ETSI TS 129 109 V8.1.0 (2009-04)

Technical Specification

Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
LTE;
Generic Authentication Architecture (GAA);
Zh and Zn Interfaces based on the Diameter protocol;
Stage 3
(3GPP TS 29.109 version 8.1.0 Release 8)

Reference
RTS/TSGC-0429109v810

Keywords
GSM, LTE, UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present stage 3 specification defines the Diameter based implementation for bootstrapping Zh interface (BSF-HSS) and Dz interface (BSF-SLF) for HSS resolution for the BSF, the MAP based implementation for bootstrapping Zh' interface (BSF-HLR) and GAA Application Zn interface (BSF-NAF) in Generic Authentication Architecture (GAA). This specification also defines the Web Services based implementation for GAA Application Zn reference point (BSF-NAF). The definition contains procedures, message contents and coding. The procedures for bootstrapping and usage of bootstrapped security association are defined in 3GPP TS 33.220 [5].

The present document also specifies the Diameter and Web Services based implementation for the GAA Application Push Function Zpn reference point (BSF-NAF). The procedures for bootstrapping are defined in 3GPP TS 33.223 [23].

This specification is a part of the Generic Authentication Architecture (GAA) specification series.

The diameter based implementation for the Zh interface is based on re-usage of Cx interface Multimedia-Auth-Request/Answer messages originally between CSCF and HSS. These messages are defined in 3GPP TS 29.229 [3]. The 3GPP IMS mobility management uses the same definitions between CSCF and HSS. The present document defines how the defined messages are used with the bootstrapping and GAA application procedures (e.g. subscriber certificates) and the application logic that is needed in GAA network elements (BSF, HSS, and NAF).

Figure 1.1 depicts the relationships of these specifications to the other specifications for the Diameter based implementations.

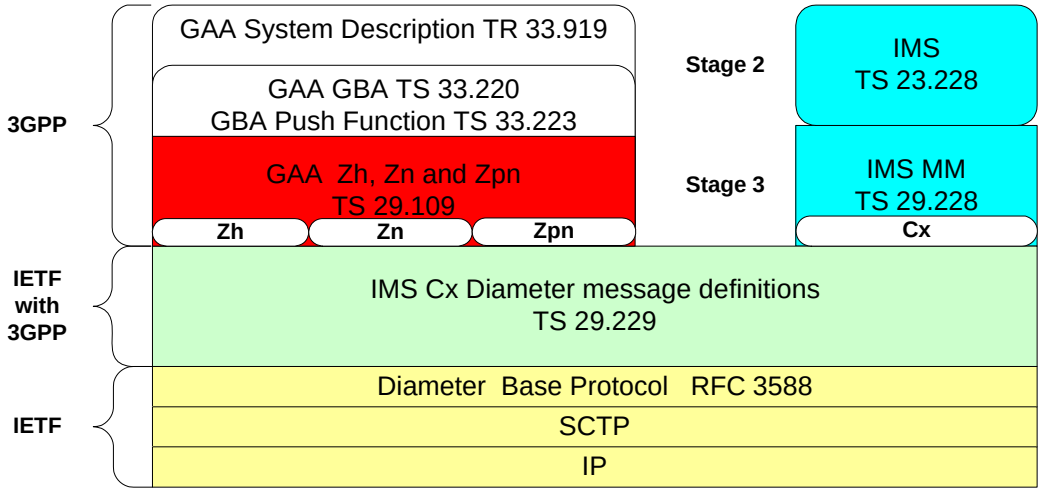


Figure 1.1: Relationships to other specifications

Figure 1.2 provides an informal overall quick introduction to the whole signalling procedures in GAA system. The important identifiers are marked bold and optional data items are italicised. The Ub and Ua interfaces, not defined in this TS, are simplified.

NOTE: The Zh' interface (BSF-HLR) is not represented in this figure.

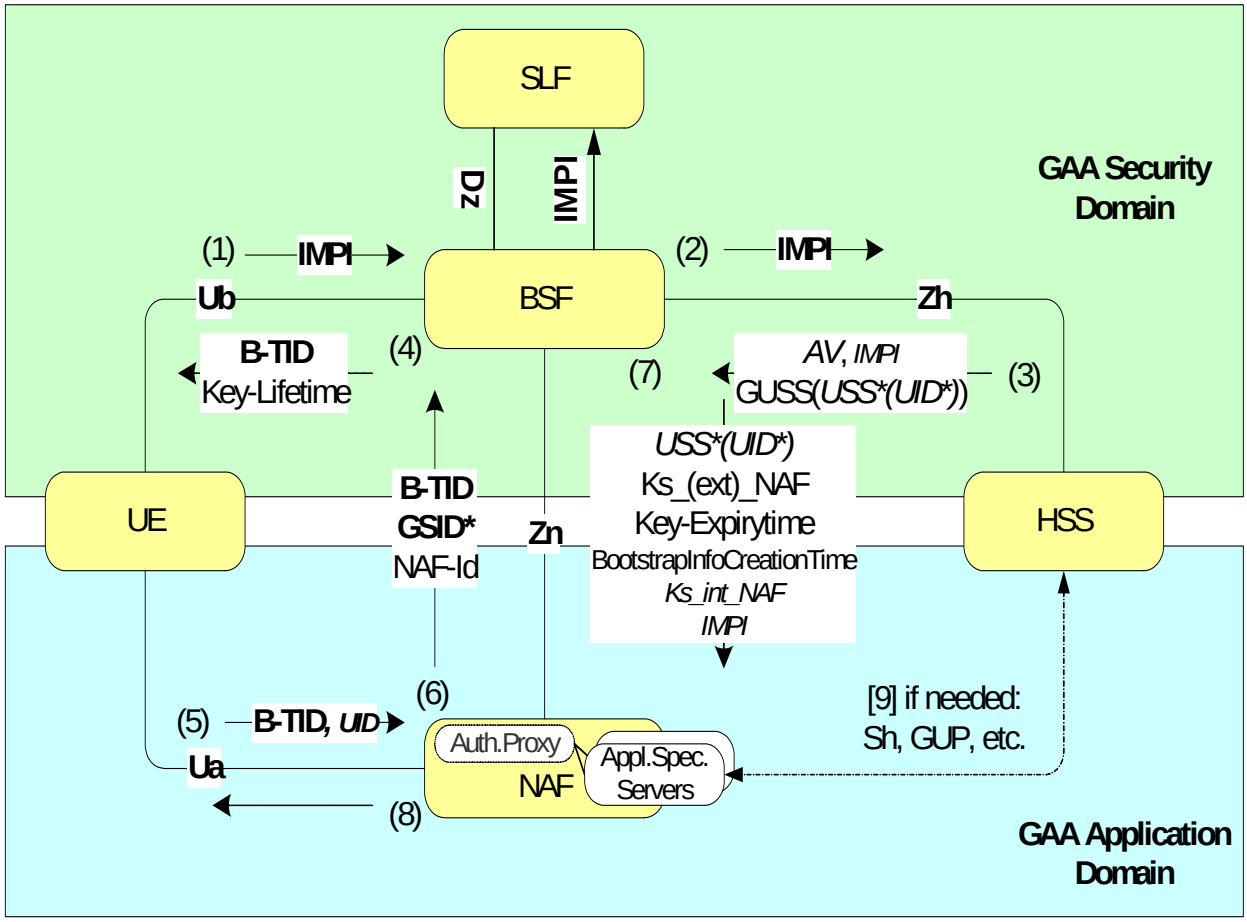
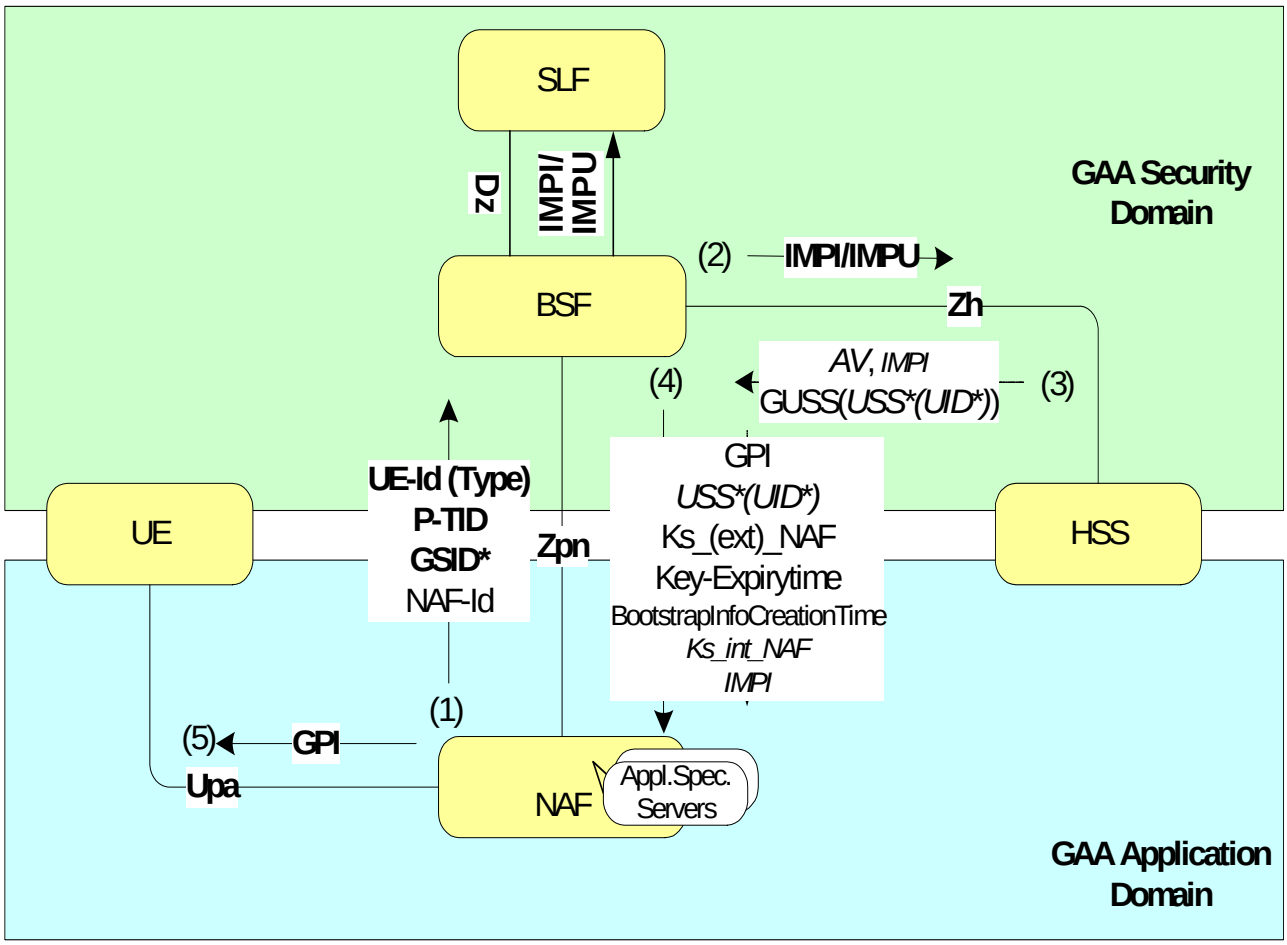


Figure 1.2: The whole signalling procedure in GAA system

Figure 1.3 provides an informal overall quick introduction to the whole signalling procedures in GAA Push Function. The important identifiers are marked bold and optional data items are italicised. The Ua and Upa interfaces, not defined in this TS, are simplified.



Bold=Important Identity. *Italic*=optional items. Ub and Ua interfaces are simplified.

Figure 1.3: Signalling procedure in GAA Bootstrapping Push Function

2 References

The following documents contain provisions that, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] IETF RFC 3588, "Diameter Base Protocol".

[2] 3GPP TS 29.228: "IP Multimedia (IM) Subsystem Cx and Dx Interfaces; Signalling flows and message contents".

[3] 3GPP TS 29.229: "Cx and Dx interfaces based on the Diameter protocol".

[4] 3GPP TR 33.919 "Generic Authentication Architecture (GAA); System Description".

[5] 3GPP TS 33.220 "Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture".

[6] 3GPP TS 33.221 "Generic Authentication Architecture (GAA); Support for Subscriber Certificates".

[7] 3GPP TS 24.109: "Bootstrapping interface (Ub) and Network application function interface (Ua);Protocol details".

[8] 3GPP TS 29.230: "Diameter applications; 3GPP specific codes and identifiers"

[9] IETF RFC 3589: "Diameter Command Codes for Third Generation Partnership Project (3GPP)".

[10] 3GPP TS 23.008: "Organisation of subscriber data"

[11] 3GPP TS 33.222: "Generic Authentication Architecture (GAA); Access to network application functions using secure hypertext transfer protocol (HTTPS)".

[12] 3GPP TS 23.228: "IP Multimedia Subsystem (IMS); Stage 2"

[13] W3C: "Web Services Activity", http://www.w3.org/2002/ws/.

[14] W3C: "Web Services Description Language (WSDL) Version 2.0 Part 0: Primer", http://www.w3.org/TR/2005/WD-wsdl20-primer-20050803/.

[15] 3GPP TR 33.980: "Liberty Alliance and 3GPP Security Interworking; Interworking of Liberty Alliance ID-FF, ID-WSF and Generic Authentication Architecture".

[16] Liberty Alliance Project: "Liberty ID-FF Authentication Context Specification".

[17] 3GPP TS 33.110: "Key establishment between a Universal Integrated Circuit Card (UICC) and a terminal"

[18] 3GPP TS 33.259: "Key establishment between a UICC Hosting Device and a Remote Device"

[19] 3GPP TS 29.002: "Mobile Application Part (MAP) Specification"

[20] 3GPP TS 33.102: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture".

[21] 3GPP TS 23.003: "Numbering, addressing and identification".

[22] OASIS Standard: "Authentication Context for the OASIS Security Assertion Markup Language (SAML) V2.0 OASIS Standard, 15 March 2005, saml-authn-context-2.0-os".

[23] 3GPP TS 33.223 "Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA) Push Function".

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TS 23.008 [10], 3GPP TR 33.919 [4], 3GPP TS 33.220 [5] apply with following additions.

Bootstrapping information (Bootstrapped data) in a BSF consists of a bootstrapping transaction identifier (B-TID), a key material (Ks), the key lifetime (expiry time), the boostrapinfo creation time, the IMPI and the GUSS (if received from HSS) with BSF control information. Each bootstrapping procedure creates a bootstrapped data entity with B-TID as retrieval key..

GAA application is an application that uses the security association created by GBA Bootstrapping procedure.

GAA service is an operator specific end user service that uses the security association created by GAA Bootstrapping procedure. GAA services are identified by **GAA Service Identifiers**. A GAA service is implemented using some standardised or propriatary GAA application defined by GAA application type.

NAF specific Bootstrapping information transferred from a BSF to a NAF contains NAF and its service specific parts from bootstrapped data and needed key information derived from the bootstrapped data.

Service/Application. The term service is used here in its common meaning. A service is something that a MNO offers to subscribers. GAA Services are identified by GAA Service Identifier (GSID). In stage 2 documents ([4], [5], [6] and [11]) the term application is used in the same meaning i.e. MNOs offer applications to subscribers. There is a reason to avoid the usage of the term application here. The application is an already reserved term in Diameter. In Diameter applications are identified by Application Identifiers.

3.2 Symbols

For the purposes of the present document, the terms and definitions given in 3GPP TS 23.008 [10].

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AK	Anonymity Key
AKA	Authentication and Key Agreement
AUTN	Authentication token
AV	Authentication Vector. 3GPP AV=[RAND,AUTN,XRES,CK,IK].
AVP	Attribute-Value-Pair in Diameter messages.
BIA	BootstrappingInfo-Answer message
BIR	BootstrappingInfo-Request message
BS	BootStrapping Procedure
BSF	Bootstrapping server functionality BSF is hosted in a network element under the control of an MNO.
B-TID	Bootstrapping Transaction Identifier
CA	Certificate Authority
CK	Confidential Key
FQDN	Full Qualified Domain Name in URI (e.g. http://FQDN:80)
GAA	Generic Authentication Architecture
GBA	Generic Bootstrapping Architecture
GPI	GBA Push Information
GSID	GAA Service Identifier
GUSS	GBA User Security Settings
HSS	Home Subscriber System
IK	Integrity Key
IMPI	IP Multimedia Private Identity

IMPU	IP Multimedia Public Identity
Ks	Key Material
Ks_ext_NAF	MEbased key for a specific NAF
ME	Mobile Equipment
MNO	Mobile network operator
NAF	Operator-controlled network application function functionality. NAF is hosted in a network element under the control of an MNO.
RAND	Random challenge in authentication
REQ	In Diameter header indicates that the message is a Request.
SCTP	Stream Control Transmission Protocol
SLF	Subscription Location Function
SSC	Subscriber Certificate Procedure
Ua	UE-NAF interface for GAA applications
Ub	UE-BSF interface for bootstrapping
UE	User Equipment
Ks_int_NAF	UICC based key for a specific NAF
USS	User Security Settings (a part of GUSS)
XRES	Expected response in authentication
Zh	BSF-HSS interface for bootstrapping procedure
Zh'	BSF-HLR interface for bootstrapping procedure
Zn	BSF-NAF interface for GAA applications.

4 GBA Bootstrapping Zh interface and Zh' interface

4.1 Generic bootstrapping network architecture

The network architecture of the Diameter based implementation for Bootstrapping procedure is presented in Figure 4.1-1. The interface Ub (bootstrapping) is defined in 3GPP TS 24.109 [7] and the interface Zh in this specification.

Figure 4.1-1: Network architecture of bootstrapping procedure

The generic boostrapping network architecture for Bootstrapping Push procedures is presented in Figure 4.1-2. The interface Zpn is also defined in this specification.

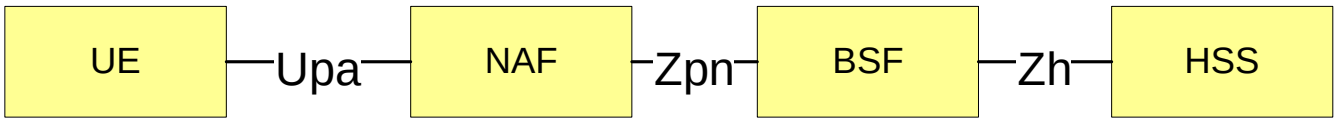


Figure 4.1-2: Network architecture of bootstrapping push procedure

The protocol stack of the Zh interface in Bootstrapping procedure is presented in Figure 4.2. The Diameter Base protocol is defined in [1] and the Diameter application in 3GPP TS 29.229 [3]. The requirements for Zh interface are defined in 3GPP TS 33.220 [5] and 3GPP TS 33.223 [23].

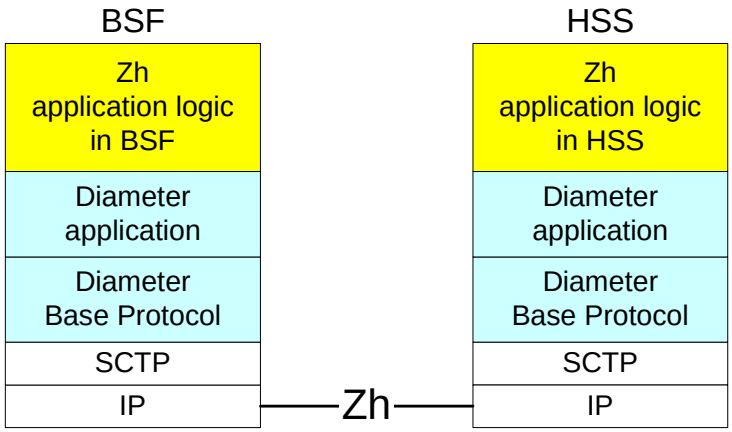


Figure 4.2: Protocol stack of Zh interface

The MAP based Bootstrapping Zh' interface is defined in the section 4.3.

4.2 Protocol Zh between BSF and HSS

The requirements for Zh interface are defined in 3GPP TS 33.220 [5] and 3GPP TS 33.223 [23].

The Bootstrapping Zh interface performs the retrieval of an authentication vector and possibly GBA User Security Settings from the HSS.

The overall Bootstrapping procedure is depicted in Figure 4.3. The basic procedure is:

A) A UE starts the bootstrapping procedure by protocol Ub with a BSF giving the IMPI of the user (see 3GPP TS 24.109 [7]).

B) The BSF starts protocol Zh with user"s HSS

- The BSF requests user"s authentication vector and GBA User Security Settings(GUSS) corresponding to the IMPI.
- The HSS supplies to the BSF the requested authentication vector and GUSS (if any).

NOTE: If there is more than one HSS deployed within the network, the BSF may have to contact the SLF using the Dz interface prior to sending the request for information to the HSS (see section 6.4).

C) The BSF continues the protocol Ub with the UE (see 3GPP TS 24.109 [7]).

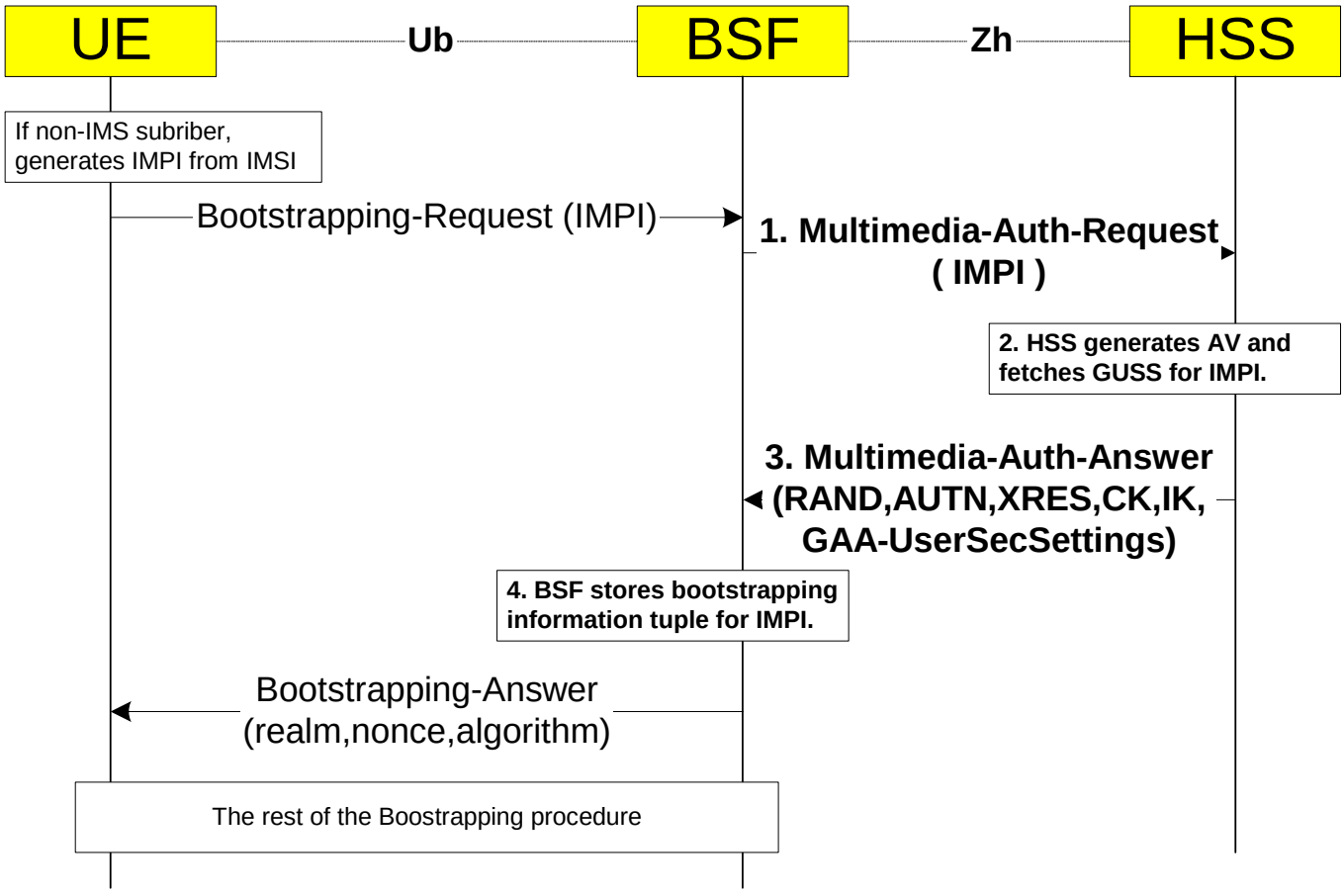


Figure 4.3: The GBA bootstrapping procedure

The overall Bootstrapping Push procedure is depicted in Figure 4.4. The Push procedure is:

A) A NAF sends a bootstrapping push request to the BSF via Zpn interface including the IMPI or IMPU of the user (see section 5).

B) The BSF starts protocol Zh with user"s HSS

- The BSF requests user"s authentication vector and GBA User Security Settings (GUSS) corresponding to the IMPI or IMPU.
- The HSS supplies to the BSF the requested authentication vector and GUSS (if any).

NOTE: If there is more than one HSS deployed within the network, the BSF may have to contact the SLF using the Dz interface prior to sending the request for information to the HSS (see section 6.4).

C) The BSF continues the protocol Zpn with the NAF (see section 5).

D) The NAF starts protocol Upa with the UE, as specified in 3GPP TS 24.109 [7].

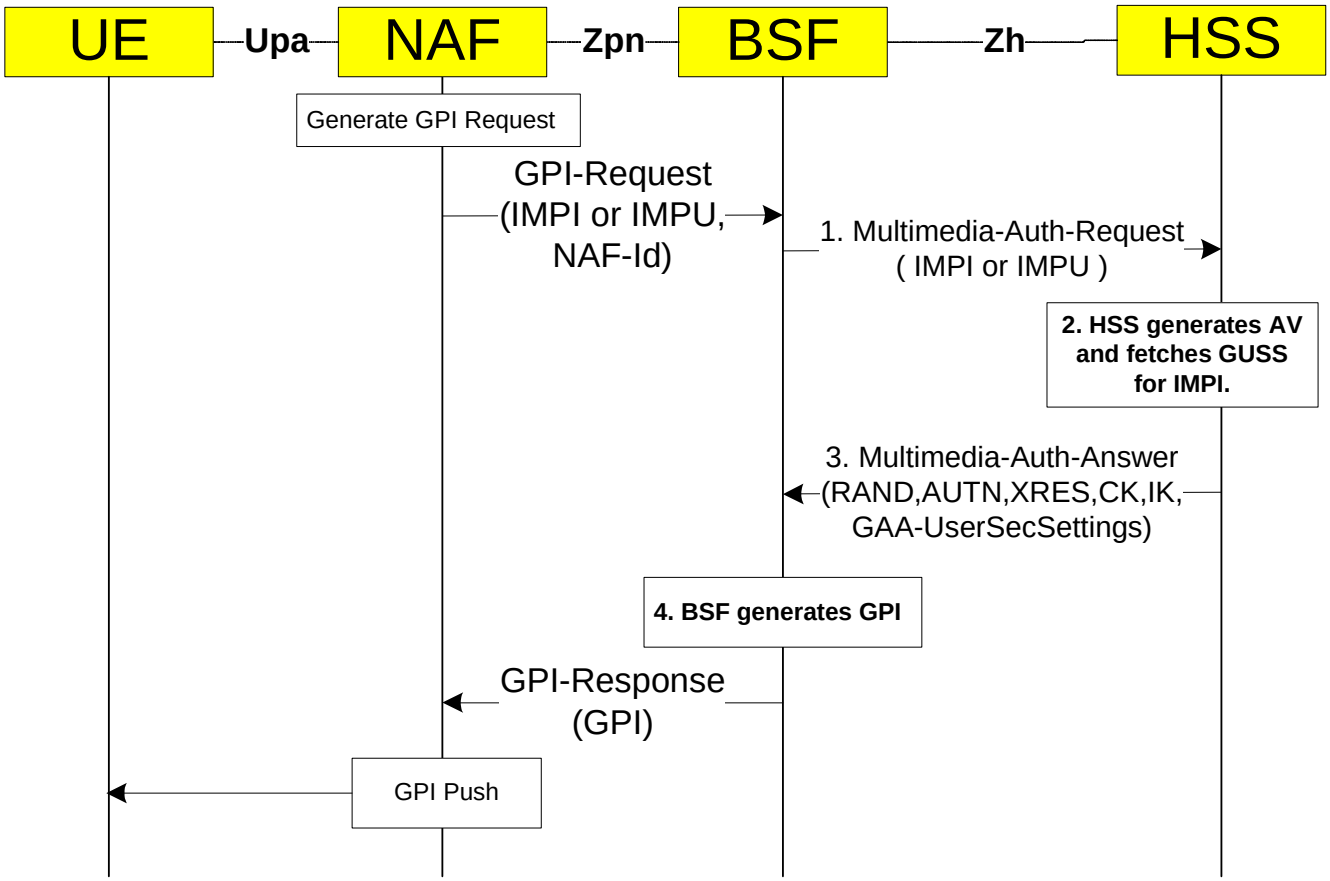


Figure 4.4: The GBA bootstrapping Push procedure

The steps of the bootstrapping procedure in Figure 4.3 are:

Step 1

The BSF shall send the following Bootstrapping-Request to the HSS in the format of Multimedia-Auth-Request (MAR) message. The content of the message is given below in the same format as in 3GPP TS 29.229 [3]. The curly brackets indicate mandatory AVPs. The square brackets indicate optional AVPs. The "address of" refers to the Fully Qualified Host Name (FQDN).

```
<Multimedia-Auth-Request> ::=<Diameter Header: 303, REQ, PXY, 16777221 >
                < Session-Id >
                { Vendor-Specific-Application-Id }
                { Auth-Session-State }            ; NO_STATE_MAINTAINED
                { Origin-Host }                   ; Address of BSF
                { Origin-Realm }                  ; Realm of BSF
                { Destination-Realm }             ; Realm of HSS
                [ Destination-Host ]              ; Address of the HSS
                [ User-Name ]                     ; IMPI from UE
                [ Public-User-Identity ]          ; IMPU from UE
                [ GUSS-Timestamp ]                ; Timestamp of GUSS in BSF
                *[ AVP ]
                *[ Proxy-Info ]
                *[ Route-Record ]
```

The content of mandatory Vendor-Specific-Application-ID according [1] is:

```
<Vendor-Specific-Application-Id>::=<AVP header: 260>
                1*  [Vendor-Id]                    ; 3GPP is 10415
                0*1 {Auth-Application-Id}          ; 16777221
                0*1 {Acct-Application-Id}          ; Omitted
```

When determining the value of Destination-Host AVP the BSF can use redirector function (SLF) to resolve the address of the HSS if needed (see 3GPP TS 29.229 [3]). The BSF shall set the Auth-Session-State AVP to NO_STATE_MAINTAINED to inform that the HSS does not need to maintain any status information for this session according 3GPP TS 29.229 [3].

The User-name is the IMS Private User Identity (IMPI) as required in 3GPP TS 29.228 [2]. In bootstrapping push procedures the IMPI may not be available to the NAF and BSF. In such cases, the Public-User-Identity AVP will be present instead, and will contain the IMS Public User Identity, otherwise only the User-Name shall be present.

If the BSF supports the GUSS timestamp mechanism and has local copy of the GUSS, which has a timestamp, the BSF may include the GUSS-Timestamp AVP. In this case the GUSS-Timestamp AVP shall contain the timestamp from subscriber's GUSS. Otherwise the GUSS-Timestamp AVP shall not be present.

Step 2

When the HSS receives the MAR message, the HSS shall derive the user Authentication Vector (AV) information according the IMPI and populates it into SIP-Auth-Data AVP as defined for IMS-AKA authentication scheme in 3GPP TS 29.229 [3]. If the Public User Identity is received, the HSS shall resolve to the corresponding IMPI associated to such IMPU before deriving the AV information. If IMPI to IMPU identity resolution is not possible i.e. due to the IMPU is shared with multiple IMPIs, the HSS shall set the Experimental-Result-Code to DIAMETER_ERROR_OPERATION_NOT_ALLOWED.

If GUSS exists for the IMPI, the HSS shall do one of the following:

1. If the HSS supports the GUSS timestamp mechanism and received the GUSS-Timestamp AVP in MAR message then it shall compare the timestamp of the GUSS in the HSS with the received timestamp.

   - If timestamps are equal, then it shall populate the GBA-UserSecSettings AVP with static string "GUSS TIMESTAMP EQUAL".

   - If the GUSS-Timestamp AVP was not received, or timestamps are not equal, then it shall populate the GBA-UserSecSettings AVP with the GUSS.

2. If the HSS does not support GUSS timestamp mechanism, it shall populate the GBA-UserSecSettings AVP with the GUSS.

The MAR/MAA sequence in the Zh interface must not change possible status information of the possible simultaneously ongoing IMS MM application sessions in the HSS.

If the User-Name (IMPI) or Public-User-Identity (IMPU) from the BSF is totally unknown to the HSS, the error situation 5401 is raised.

Step 3

The HSS shall send the following Bootstrapping-Answer message in the format of Multimedia-Auth-Answer (MAA) message back to the BSF.

```
< Multimedia-Auth-Answer> ::= < Diameter Header: 303, PXY, 16777221 >
                < Session-Id >
                { Vendor-Specific-Application-Id }
                [ Result-Code ]
                [ Experimental-Result]
                { Auth-Session-State }             ; NO_STATE_MAINTAINED
                { Origin-Host }                    ; Address of HSS
                { Origin-Realm }                   ; Realm of HSS
                [ User-Name ]                      ; IMPI
```

```
[ Public-Identity ]                ; IMPU
[ SIP-Auth-Data-Item ]
[ GBA-UserSecSettings ]            ; GUSS
*[ AVP ]
*[ Proxy-Info ]
*[ Route-Record ]
```

The HSS shall set the mandatory Auth-Session-State AVP to NO_STATE_MAINTAINED because the HSS does not maintain any state information about this session and the BSF does not need to send any session termination request 3GPP TS 29.229 [3].

Is the Private User Identity was present in the request, both Public-User-Identity and User-Name AVPs shall be present in the response. Otherwise, the User-name AVP (IMPI) may be sent back for checking.

The required authentication vectors are sent in the SIP-Auth-Data-Items AVP as defined for IMS-AKA authentication scheme according to 3GPP TS 29.228 [2]. The security settings of user"s all GAA applications are sent in GBA-UserSecSettings AVP.

If the 2G GBA option (see 3GPP TS 33.220, Annex I [5]) is applied for the user the SIP-Auth-Data-Item AVP shall be filled as follows: The SIP-Authentication-Scheme AVP is set to "Digest-AKAv1-MD5-2G-GBA" to indicate a 2G GBA vector for GBA. The SIP-Authenticate AVP contains only RAND. The SIP-Authorization AVP contains RES. The Confidentiality-Key AVP contains Kc. The Integrity-Key AVP shall not be present.

Step 4.

When the BSF receives the MAA message, the BSF shall check the value of the SIP-Authentication-Scheme AVP. If the BSF does not support the authentication-scheme the BSF shall stop processing the message and should indicate an error via the O&M subsystem.

The BSF generates the needed key material (Ks) from confidential key (CK) and integrity key (IK) as described in 3GPP TS 33.220 [5] and stores temporarily the tuple <IMPI,Ks,GBA-UserSecSettings> for further use in GAA applications. The rest of the bootstrapping procedure in Ub interface will later add also the Bootstrapping Transaction Identifier (B-TID) to that tuple as key and the key lifetime (expiry time).

If the BSF sent the GUSS-Timestamp AVP in step 1, and if the GBA-UserSecSettings AVP contains a static string "GUSS TIMESTAMP EQUAL", then the local copy of the GUSS in the BSF shall be preserved. If the GBA-UserSecSettings AVP was not present in the MAA message, the local copy of the GUSS shall be deleted. If the GBA-UserSecSettings AVP contains a new GUSS, the local copy of the GUSS shall be deleted, and the new GUSS shall be stored in the BSF.

In GBA Bootstrapping Push procedures, the BSF generates the requested NAF-key(s) according to provided NAF_Id and the GBA Push Information (GPI). The BSF completes the rest of the bootstrapping push procedure sending its response to the NAF over Zpn interface, and deleting the Ks used as defined in section 5.

4.3 Protocol Zh' between BSF and HLR

The requirements for Zh' reference point are the same with the one for the Zh reference point defined in 3GPP TS 33.220 [5], except that the GBA User Security Settings (GUSS) are not supported.

The Bootstrapping Zh' interface performs the retrieval of an authentication vector. The basic procedure is:

A) A UE starts the bootstrapping procedure by protocol Ub with a BSF giving the IMPI of the user (see 3GPP TS 24.109 [7]). In bootstrapping push procedures a NAF initiates a GBA Push Information Request by protocol Zpn with the BSF giving the user identity (see section 5). The BSF derives the IMSI from the IMPI. In bootstrapping push procedures when only the Public Identity of the user is made available to the BSF, the BSF shall perform public to private identity resolution as defined in section 4.3.1.

B) The BSF starts protocol Zh' with user"s HLR

- The BSF requests user"s authentication vector corresponding to the IMSI.
- The HLR supplies to the BSF the requested authentication vector.

C) The BSF continues the protocol Ub with the UE (see 3GPP TS 24.109 [7]).

The BSF shall use the following Bootstrapping-Request to the HLR in the format of a MAP_SEND_AUTHENTICATION_INFO. The content of the request, result and error messages is given below in the same format as in 3GPP TS 29.002 [19]. The parameter usage is as follows:

IMSI

See 3GPP TS 29.002 [19] section 7.6.2 for the use of this parameter.

Number of requested vectors

The BSF shall only request one authentication vector at a time.

Requesting node type

BSF

Re-synchronisation Info

For definition and use of this parameter see 3GPP TS 33.102 [20].

Segmentation prohibited indicator

This parameter is not applicable and should not be sent by the BSF.

Immediate response preferred indicator

This parameter indicates that one of the requested authentication vectors is requested for immediate use in the BSF.

Requesting PLMN ID

The PLMN-ID of the requesting node. See 3GPP TS 23.003 [21].

AuthenticationSetList

One authentication vector is transferred from the HLR to the BSF, if the outcome of the service was successful.

User error

One of the following error causes defined in 3GPP TS 29.002 [19] shall be sent by the user in case of unsuccessful outcome of the service, depending on the respective failure reason:

- unknown subscriber;
- unexpected data value;
- system failure;
- data missing.

Provider error

See 3GPP TS 29.002 [19] for the use of this parameter.

When the BSF receives the response of the MAP_SEND_AUTHENTICATION_INFO service, then it shall generate the needed key material (Ks) from confidential key (CK) and integrity key (IK) as described in 3GPP TS 33.220 [5], respecively from Kc as decribed in 3GPP TS 33.220 [5] and stores temporarily the tuple <IMPI,Ks> for further use in GAA/GBA applications. The rest of the bootstrapping procedure in Ub interface will later add also the Bootstrapping Transaction Identifier (B-TID) to that tuple as key and the key lifetime (expiry time).

4.3.1 Public to Private Identity Resolution over Zh between BSF and HLR

When the UE identity used by the NAF in a GPI Request towards the BSF is an MSISDN, the BSF shall resolve the corresponding private user identity (IMSI) for the user.

When Zh interface between BSF and HLR is used, the BSF shall resolve the private user identity with the HLR in the format of a MAP_SEND_IMSI or a MAP_SEND_ROUTING_INFO_FOR_SM.

The content of the request, result and error messages is given below in the same format as in 3GPP TS 29.002 [19]. The parameter usage when the MAP_SEND_IMSI operation is used is as follows:

MSISDN

See 3GPP TS 29.002 [19] section 7.6.2 for the use of this parameter.

IMSI

See 3GPP TS 29.002 [19] section 7.6.2 for the use of this parameter. The presence of this parameter is mandatory in a successful case.

User error

Only one of the following values is applicable:

- Unknown subscriber;
- Unexpected data value;
- Data missing.

The parameter usage when the MAP_SEND_ROUTING_INFO_FOR_SM operation is used is as follows:

MSISDN

See 3GPP TS 29.002 [19] section 7.6.2 for the use of this parameter.

SM-RP-PRI

BSF shall set this parameter to indicate that delivery of the short message shall NOT be attempted when a service centre address is already contained in the Message Waiting Data file.

Service Centre Address

BSF shall include the BSF address within this parameter.

SM-RP-MTI
SM-RP-SMEA
GPRS Support Indicator

BSF shall not use these parameters in the public to private identity resolution request.

SM-Delivery Not Intended

BSF shall set this parameter to indicate that delivery of a short message is not intended and that only IMSI is requested.

IMSI

See 3GPP TS 29.002 [19] section 7.6.2 for the use of this parameter. The presence of this parameter is mandatory in a successful case.

Network Node Number
LMSI
GPRS Node Indicator
Additional Number

BSF ignores these parameters if present in the public to private identity resolution response.

User error

One of the following error causes defined in 3GPP TS 29.002 [19] shall be sent by the user in case of unsuccessful outcome of the service, depending on the respective failure reason:

- Unknown subscriber;
- Call Barred;

- Teleservice Not Provisioned;
- Absent Subscriber_SM;
- Facility Not Supported;
- System failure;
- Unexpected Data Value;
- Data missing.

Provider error

See 3GPP TS 29.002 [19] for the use of this parameter.

When the HLR is able to resolve the private identity of the user, the BSF continues with the boostrapping procedures over Zh and requests a user''s authentication vector corresponding to the IMSI as defined above. Otherwise the BSF provides the corresponding error to the NAF over Zpn as defined in section 5.

5 GAA Application Zn and Zpn interfaces

5.1 Applications" network architecture

The network architecture of the GAA applications procedure using Zn is presented in Figure 5.1-1. The 3GPP GAA applications are listed in annex B. Different GAA applications may implement the Ua interface in different way. The Zn interface is defined in this specification.

Figure 5.1-1: Network architecture of GAA application using Zn

The network architecture of GBA push procedure using Zpn is presented in Figure 5.1-2. The Zpn interface is also defined in this specification.

Figure 5.1-2: Network architecture of GBA push procedure using Zpn

Two options are specified for the protocol stack to be used on the Zn and Zpn interface:

The protocol stack of the DIAMETER based Zn interface,

the protocol stack of the Web Services based Zn interface.

The BSF shall support both options whereas the NAF shall support one or both of these options.

The protocol stack of the Diameter based implementation of the Zn interface for GAA applications and Zpn interface for GBA push procedure is presented in Figure 5.2-1. The Zn interface is specified in clause 5.2. The Zpn interface is specified in clause 5.4. The diameter Base protocol is defined in [1] and the Diameter application in 3GPP TS 29.229 [3]. The requirements for Zn interface are defined in 3GPP TS 33.220 [5]. The requirements for Zpn interface are defined in 3GPP TS 33.223 [23].

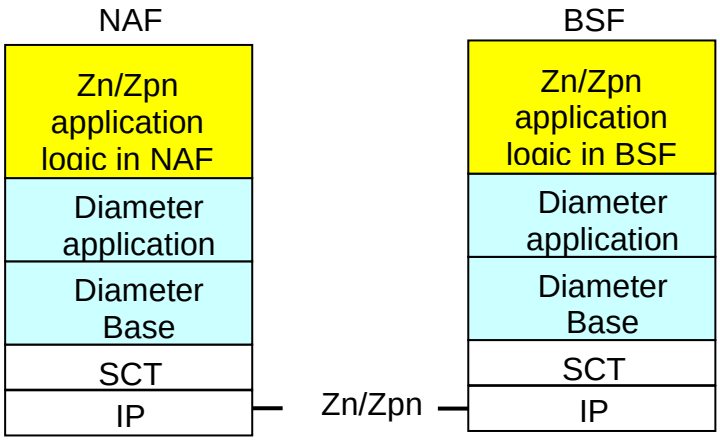


Figure 5.2-1: Protocol stack of Diameter based Zn and Zpn interfaces

The protocol stack of the Web Services based Zn interface for GAA applications and Zpn interface for GBA push procedure is presented in Figure 5.2-2. The Zn interface is specified in clause 5.3. The Zpn interface is specified in clause 5.5.

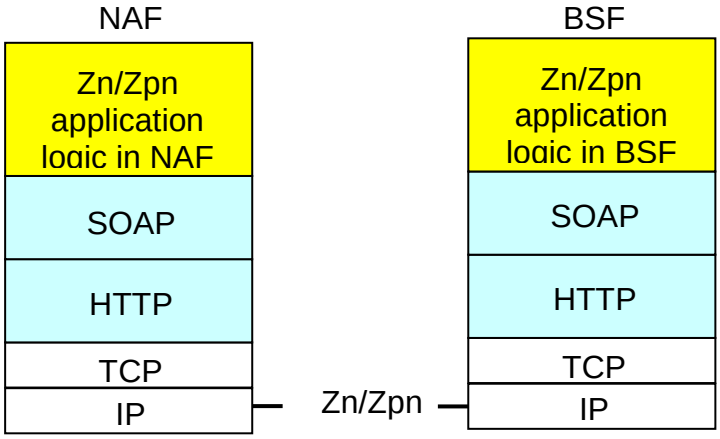


Figure 5.2-2: Protocol stack of Web Services based Zn interface

5.2 Protocol Zn between NAF and BSF based on Diameter

The requirements for Zn interface are defined in 3GPP TS 33.220 [5].

The protocol Zn retrieves the key material and possibly user security settings data by NAF from BSF. After UE is authenticated with the BSF, every time the UE wants to interact with an NAF the following steps are executed as depicted in Figure 5.3. The basic procedure is:

A) The UE starts protocol Ua (see 3GPP TS 33.220 [5])

- In general, the UE and the NAF will not yet share the key(s) required to protect protocol Ua. If they already do, there is no need for the NAF to invoke protocol Zn.
- It is assumed that UE supplies sufficient information to NAF, i.e. the Bootstrapping Transaction Identifier (B-TID), to allow the NAF to retrieve specific key material (e.g. Ks_NAF in the case of GBA_ME, and Ks_ext_NAF or Ks_int_NAF or both in the case of GBA_U) from BSF.

- The UE derives the keys required to protect protocol Ua from the key material.

B) The NAF starts protocol Zn with BSF

- The NAF requests NAF specific key material corresponding to the information supplied by the UE to the NAF (i.e. the bootstrapping transaction identifier) in the start of protocol Ua.
- The BSF generates and supplies to the NAF the requested NAF specific key material, the expiry time, the bootstrapinfo creation time, and the appropriate User Security Settings defined for received application identifiers.

C) The NAF continues protocol Ua with the UE (see 3GPP TS 33.221 [6])

Once the run of protocol Ua is completed the purpose of bootstrapping is fulfilled as it enabled UE and NAF to run protocol Ua in a secure way.

The common GAA application procedure is presented in Figure 5.3.

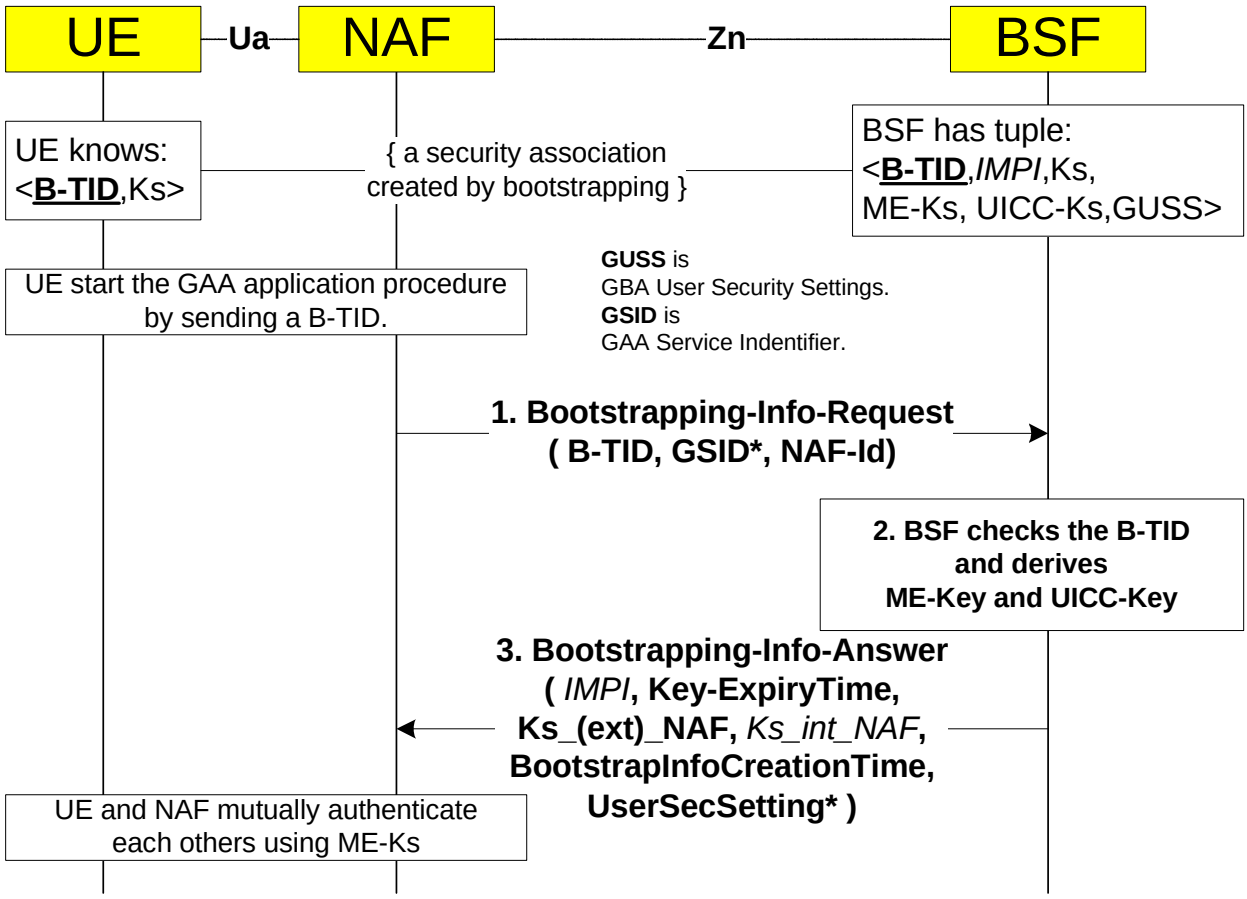


Figure 5.3: The Diameter based GAA application procedure

The steps of the GAA application procedure in Figure 5.3 are:

Step 1

The NAF shall send a Bootstrapping-Info-Request message (BIR) to the BSF. The content of the message is given here in the same format as in 3GPP TS 29.229 [3]. The curly brackets indicate mandatory AVPs. The square brackets indicate optional AVP. The address refers to the Fully Qualified Host Name (FQDN).

```
< Bootstrapping-Info-Request> ::=<Diameter Header: 310, REQ, PXY, 16777220 >
                < Session-Id >
                { Vendor-Specific-Application-Id }
                { Origin-Host }                          ; Address of NAF
                { Origin-Realm }                         ; Realm of NAF
                { Destination-Realm }                    ; Realm of BSF
                [ Destination-Host ]                     ; Address of the BSF
                * [ GAA-Service-Identifier ]             ; Service identifiers
                { Transaction-Identifier }               ; B-TID
                { NAF-Id }                               ; NAF_ID
                [ GBA_U-Awareness-Indicator ]            ; GBA_U awareness of the NAF
                *[ AVP ]
                *[ Proxy-Info ]
                *[ Route-Record ]
```

The content of Vendor-Specific-Application-ID according [1] is:

```
<Vendor-Specific-Application-Id>::=<AVP header: 260>
                1*  [Vendor-Id]                          ; 3GPP is 10415
                0*1 {Auth-Application-Id}                ; 16777220
                0*1 {Acct-Application-Id}                ; Omitted
```

The Destination-Realm AVP is set to subscriber"s BSF. The address of the BSF is extracted from the B-TID.

NOTE: In the case where the subscriber has contacted a NAF that is in a visited network, the NAF contacts the subscriber's home BSF through a Diameter based Proxy (Zn-Proxy) that is located in the same network as the NAF. The local BSF and the Zn-Proxy may be co-located. See 3GPP TS 33.220 [6].

The NAF indicates the GAA services for which the information is retrieved by GAA-Service-Identifier AVPs. The Bootstrapping Transaction Identifier defines the earlier bootstrapping procedure execution.

Step 2

In the successful case the BSF has a tuple <B-TID,IMPI,Ks, Key lifetime, Bootstrapinfo creation time, GBA-UserSecSettings> identified by Bootstrapping Transaction Identifier (B-TID). When the BSF receives the request it checks the existence and validity of the tuple for given B-TID. If checking fails the BSF sends an Answer message with Experimental-Result set to indicate the error type 5403. If the tuple for B-TID exists, but is expired, error type 5403 is also send to indicate needs for renewal of the boostrapping procedure. In successful case the Result-Code is set to 2xxx as defined in [1].

The BSF derives the key material for the ME (i.e., Ks_NAF in the case of GBA_ME, and Ks_ext_NAF in the case of GBA_U) and possibly the key material for the UICC (i.e., Ks_int_NAF in the case of GBA_U) according to the B-TID and packs them into ME-Key-Material AVP and possible UICC-Key-Material AVP. The ME-Key-Material contains Ks_(ext)_NAF and the UICC-key-Material contains the Ks_int_NAF key. The BSF select correct user"s Security Settings according the request"s GAA-Service-Identifier AVP to GBA-UserSecSettings AVP. If NAF grouping is used by the operator and there are one or more USSs corresponding to the requested GSID, then also the nafGroup attribute of USS is checked. If the NAF has sent a GAA-Service-Identifier that does not have corresponding user"s security settings, and the BSF is locally configured to reject those requests from the NAF, then the error 5402 is raised. If the NAF has sent a GAA-Service-Identifier that have corresponding user's security settings, but the BSF is locally configured to reject those from that NAF, then the error 5402 is raised too.

The NAF may be addressed from the UE with different FQDNs. The BSF shall check if this NAF-Id is allowed to be used for the NAF. If the NAF identified by its Origin-Host AVP is configured in the BSF not to be authorized to use the given NAF-Id, the BSF may raise the error situation 5402. The BSF may also be configured so that a certain NAF is not authorized to use a certain GAA-Service-Identifier. This situation may be also indicated by error code 5402.

Step 3

After that the BSF shall send a Bootstrapping-Info-Answer message (BIA) back to the NAF.

```
< Boostrapping-Info-Answer> ::= < Diameter Header: 310, PXY, 16777220 >
                < Session-Id >
                { Vendor-Specific-Application-Id }
                [ Result-Code ]
                [ Experimental-Result]
                { Origin-Host }                    ; Address of BSF
                { Origin-Realm }                   ; Realm of BSF
                [ User-Name ]                      ; IMPI
                [ ME-Key-Material ]                ; Required
                [ UICC-Key-Material ]              ; Conditional
                [ Key-ExpiryTime ]                 ; Time of expiry
                [ BootstrapInfoCreationTime ]      ; Bootstrapinfo creation time
                [ GBA-UserSecSettings ]            ; Selected USSs
                [ GBA-Type ]                       ; GBA type used in bootstrapping
                *[ AVP ]
                *[ Proxy-Info ]
                *[ Route-Record ]
```

The BSF may or may not send the User-name AVP (IMPI) according its configuration.

The mandatory common key material with the ME (Ks_NAF or Ks_ext_NAF) is sent in the ME-Key-Material AVP. The common key material with the UICC (Ks_int_NAF) is optionally sent in the UICC-Key-Material AVP only if the "uiccType" tag in bsfInfo from the HSS is set to "GBA_U".

The Key-ExpiryTime AVP contains the expiry time of the Bootstrapping information in the BSF according its configuration. The expiry time is represented according the Diameter Time data format in seconds that have passed since 0h on January 1, 1900 UTC . If a special key lifetime value is given in the "lifeTime" tag inside the bsfInfo from the HSS in bootstraping procedure, it is used instead of the BSF default configuration value when the expiry time is calculated.

The BootstrapInfoCreationTime AVP contains the bootstrapinfo creation time, i.e., creation time of the Bootstrapping information in the BSF. The bootstrapinfo creation time is represented in seconds that have passed since January 1, 1900 00:00:00.000 UTC.

The BSF selects the appropriate User Security Settings (if any) to the GBA-UserSecSettings AVP from stored GAA-UserSecSettings in Bootstrapping information according the GBA-Service-Identifier AVPs in the request message.

The BSF shall indicate the type of used authentication in the bootstrapping procedure to the NAF in GBA-Type, if other than 3G GBA type has been performed.

When the NAF receives the BIA message, the NAF shall check the value of the GBA-Type AVP if it is included in the message. If the NAF does not support the GBA-Type the NAF shall stop processing the message and should indicate an error via the O&M subsystem. The further procedure in the NAF when the BIA is received is described in 3GPP TS 33.220 [5], 3GPP TS 33.222 [11] and optionally in GAA service type specific TSs.

5.3 Protocol Zn between NAF and BSF based on Web Services

The procedures in the NAF and in the BSF related Web Services [13] based Zn interface are the same as specified in clause 5.2, but instead of Diameter messages a Web Services procedures shall be used to communicate over Zn interface. Annex D specifies the GBA Service for Web Services, i.e., it contains the Web Services Definition Language (WSDL) [14] specification for GBA Service. Below are the attributes that are defined for GBA Service request, response, and fault cases.

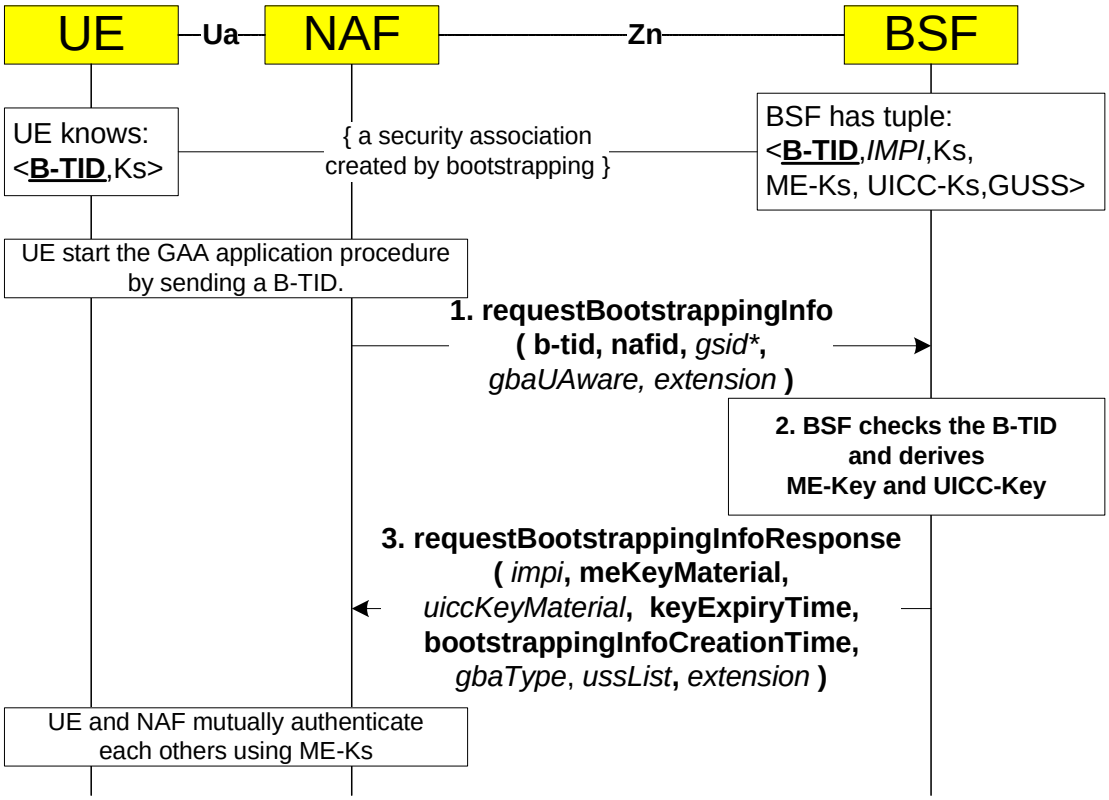


Figure 5.4: The Web Services based GAA application procedure

The possible attribute value definitions and restrictions for the request, response, and fault messages are the same as in clause 6 unless explicitly specified in this clause. The steps of the GAA application procedure in Figure 5.4 are:

Step 1

The NAF shall send a requestBootstrappingInfo message to the BSF. The schema of the message content is given here and it shall be in the same format as it is in WSDL.

```
<xsd:element name="requestBootstrappingInfoRequest">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="btid" type="xsd:string"/>
      <xsd:element name="nafid" type="xsd:base64Binary"/>
      <xsd:element name="gsid" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="gbaUAware" type="xsd:boolean" minOccurs="0"/>
      <xsd:element name="extension" type=" typens:tExtension" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

The SOAP message that shall be sent to BSF, i.e., the URI of the message shall contain the BSF address extracted from the B-TID.

In the case where the subscriber has contacted a NAF that is in a visited network, the NAF contacts the subscriber's home BSF through a GBA-Proxy that is located in the same network as the NAF. The local BSF and the GBA-Proxy may be co-located. See 3GPP TS 33.220 [6].

The NAF indicates the GAA services for which the information is retrieved by "gsid" elements. The "btid" element defines the earlier bootstrapping procedure execution. The "gbaUAware" element indicates whether NAF is GBA_U aware, and is capable of using and handling the "uiccKeyMaterial". The default value for "gbaUAware" is false. The NAF may use one or more "extension" elements to include additional data to the request, but the BSF may ignore the additional data.

Step 2

The procedures for step 2 are the same as in step 2 in clause 5.2.

Step 3

After that requestBootstrappingInfoResponse back to the NAF.

```
<xsd:element name="requestBootstrappingInfoResponse">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="impi" type="xsd:string" minOccurs="0"/>
      <xsd:element name="meKeyMaterial" type="xsd:base64Binary"/>
      <xsd:element name="uiccKeyMaterial" type="xsd:base64Binary" minOccurs="0"/>
      <xsd:element name="keyExpiryTime" type="xsd:dateTime"/>
      <xsd:element name="bootstrappingInfoCreationTime" type="xsd:dateTime"/>
      <xsd:element name="gbaType" type="xsd:string" minOccurs="0"/>
      <xsd:element name="ussList" type="xsd:string" minOccurs="0"/>
      <xsd:element name="extension" type=" typens:tExtension" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

The BSF may or may not send the "impi" element according its configuration.

The mandatory common key material with the ME (Ks_NAF or Ks_ext_NAF) is sent in the "meKeyMaterial" element. The common key material with the UICC (Ks_int_NAF) is optionally sent in the "uiccKeyMaterial" element only if the "uiccType" tag in bsfInfo from the HSS is set to "GBA_U".

The "keyExpiryTime" element contains the expiry time of the Bootstrapping information in the BSF according its configuration. If a special key lifetime value is given in the "lifeTime" tag inside the bsfInfo from the HSS in bootstraping procedure, it shall be used instead of the BSF default configuration value when the expiry time is calculated.

The "bootstrappingInfoCreationTime" element shall contain the bootstrapinfo creation time, i.e., creation time of the Bootstrapping information in the BSF.

The BSF shall select the appropriate User Security Settings (if any) into the "ussList" element from stored GAA-UserSecSettings in Bootstrapping information according "gsid" elements in the request message. The "ussList" element contains a standalone XML document whose root element shall be "ussList" element as specified in Annex A and which contains the User Security Settings selected by the BSF.

The BSF shall indicate the type of used authentication in the bootstrapping procedure to the NAF in "gbaType" element, if other than 3G GBA type has been performed.

The BSF may use one or more "extension" elements to include additional data to the request, but the NAF may ignore the additional data.

When the NAF receives the requestBootstrappingInfoResponse message, the NAF shall check the value of the "gbaType" element if it is included in the message. If the NAF does not support the GBA type the NAF shall stop processing the message and should indicate an error via the O&M subsystem. The further procedure in the NAF when the requestBootstrappingInfoResponse message is received is described in 3GPP TS 33.220 [5], 3GPP TS 33.222 [11] and optionally in GAA service type specific TSs.

5.4 Protocol Zpn between NAF and BSF based on Diameter

The requirements for Zpn interface are defined in 3GPP TS 33.223 [23].

The protocol Zpn retrieves the GPI, key material and possibly user security settings data by NAF from BSF. When the NAF wants to send data to the UE, but the NAF has no valid NAF-key available, the following steps are executed as depicted in Figure 5.4. The basic procedure is:

In general, the UE and the NAF will not yet share the key(s) required to protect protocol Ua. If they already do, there is no need for the NAF to invoke protocol Zpn.

It is assumed that the NAF has sufficient information available, i.e. user identity and the address of BSF.

A) The NAF starts protocol Zpn with BSF

- The NAF requests GBA-Push-Info (GPI), NAF specific key material corresponding to the user identity.

- The BSF fetches AVs from the HSS, generates and supplies to the NAF the requested GPI, NAF specific key material, the expiry time, the bootstrapinfo creation time, and the appropriate User Security Settings defined for received application identifiers.

B) The NAF sends GPI over protocol Upa to the UE (see 3GPP TS 33.223 [23])

Once the run of protocol Upa is completed the purpose of bootstrapping is fulfilled and it will enable the UE and NAF to run protocol Ua in a secure way.

The GBA push procedure over Zpn is presented in Figure 5.4.

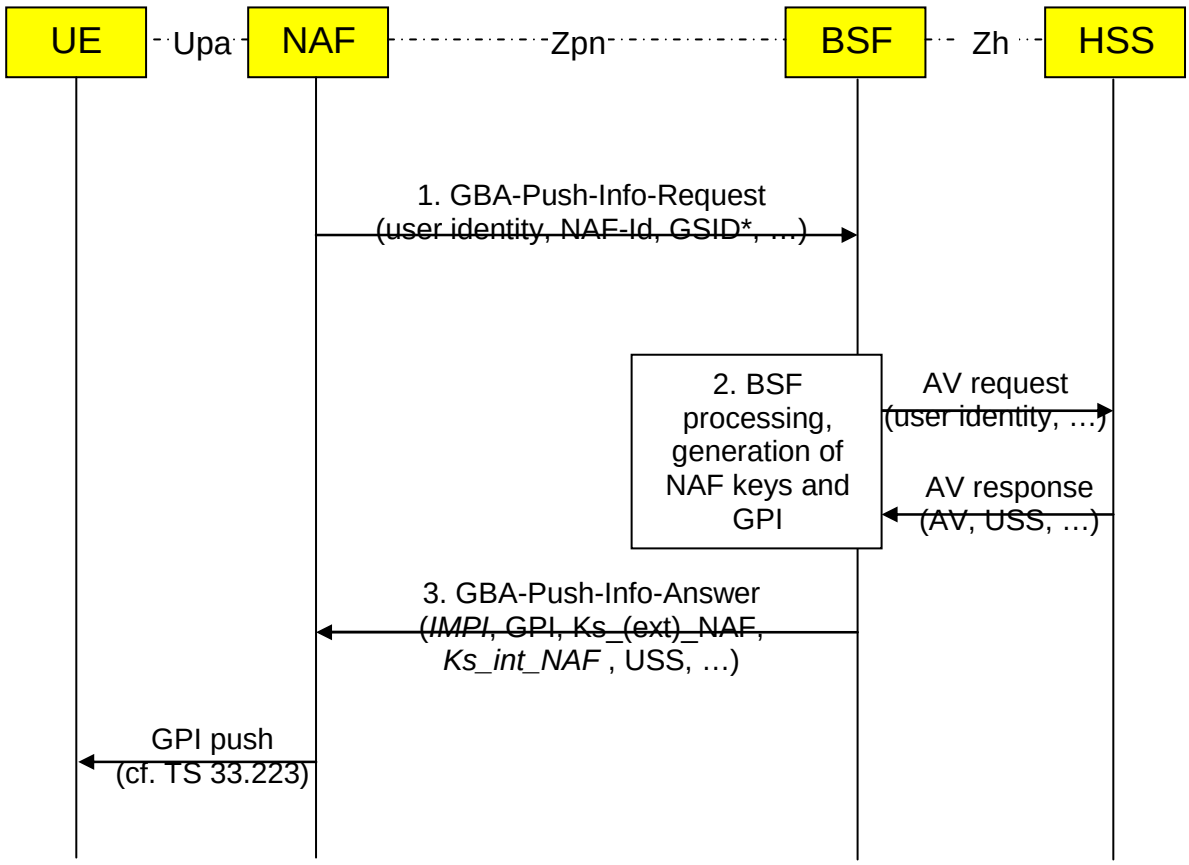


Figure 5.4: The Diameter based GBA push procedure over Zpn

The steps of the GBA push procedure in Figure 5.4 are:

- Step 1

The NAF shall send a GBA-Push-Info-Request message to the BSF. The content of the message is given here in the same format as in 3GPP TS 29.229 [3]. The curly brackets indicate mandatory AVPs. The square brackets indicate optional AVP. The address refers to the Fully Qualified Domain Name (FQDN).

```
< GBA-Push-Info-Request> ::=<Diameter Header: xxx, REQ, PXY, xxxxxxxx >
                < Session-Id >
                { Vendor-Specific-Application-Id }
                { Origin-Host }                          ; Address of NAF
                { Origin-Realm }                         ; Realm of NAF
                { Destination-Realm }                    ; Realm of BSF
                [ Destination-Host ]                     ; Address of the BSF
                { UE-Id }                                ; User identity
                { UE-Id-Type }                           ; Public or private identity
                { UICC-App-Label }                       ; UICC application label
                * [ GAA-Service-Identifier ]             ; Service identifiers
                { NAF-SA-Identifier }                    ; P-TID
                { NAF-Id }                               ; NAF_ID
                { UICC-ME }                              ; Use of GBA_ME or GBA_U
                { Requested-Key-Lifetime }               ; Requested NAF-Key lifetime
                { Private-Identity-Request }             ; Request private identity
                [ GBA_U-Awareness-Indicator ]            ; GBA_U awareness of the NAF
                *[ AVP ]
                *[ Proxy-Info ]
                *[ Route-Record ]
```

The content of Vendor-Specific-Application-ID according [1] is:

```
<Vendor-Specific-Application-Id>::=<AVP header: 260>
                1*  [Vendor-Id]                          ; 3GPP is 10415
                0*1 {Auth-Application-Id}                ; xxxxxxxx
                0*1 {Acct-Application-Id}                ; Omitted
```

The Destination-Realm AVP and Destination-Host AVP are set to subscriber"s BSF.

NOTE: In the case the NAF is in a visited network, the NAF contacts the subscriber's home BSF through a Diameter based Proxy (Zn-Proxy) that is located in the same network as the NAF. In this case the Destination-Host shall not be included.The local BSF and the Zn-Proxy may be co-located. See 3GPP TS 33.220 [6].

The NAF indicates the GAA services for which the information is retrieved by GAA-Service-Identifier AVPs.

With UE-Id and UE-Id-Type AVPs the NAF indicates the user identity and its type, i.e. private or public. The NAF uses UICC-ME and UICC-App-Label AVPs to identity which UICC application is to be used and if GBA-ME or GBA_U should be used. Private-Identity-Request indicates if he NAF requests the BSF to send the private user identity. Requested-Key-Lifetime AVP indicates the requested key lifetime for the NAF keys. NAF-SA-Identifier (P-TID) is sent to the BSF so that BSF can include it to the encrypted GPI. See 3GPP TS 33.223 [23]).

- Step 2

When the BSF receives the GPI request it checks that the requested key lifetime in the GPI request is less than the allowed max value in the system. If checking fails the BSF sends an Answer message with Experimental-Result set to indicate the error type 5402. In successful case the Result-Code is set to 2xxx as defined in [1].

The BSF includes the received user identity (public of private) to the Zh protocol request.

If GBA_U was requested by the NAF, the BSF queries its database to find out if the private user identity is registered and if a valid Ks already exists. If a valid Ks exists, the BSF shall invalidate this Ks

The BSF derives the key material for the ME (i.e., Ks_NAF in the case of GBA_ME, and Ks_ext_NAF in the case of GBA_U) and possibly the key material for the UICC (i.e., Ks_int_NAF in the case of GBA_U) according to the NAF-ID and packs them into ME-Key-Material AVP and possible UICC-Key-Material AVP. The ME-Key-Material contains Ks_(ext)_NAF and the UICC-key-Material contains the Ks_int_NAF key. The BSF select correct user"s Security Settings according the request"s GAA-Service-Identifier AVP to GBA-UserSecSettings AVP. If NAF grouping is used by the operator and there are one or more USSs corresponding to the requested GSID, then also the nafGroup attribute of USS is checked. If the NAF has sent a GAA-Service-Identifier that does not have corresponding user"s security settings, and the BSF is locally configured to reject those requests from the NAF, then the error 5402 is raised. If the NAF has sent a GAA-Service-Identifier that

have corresponding user's security settings, but the BSF is locally configured to reject those from that NAF, then the error 5402 is raised too.

The BSF generates the GPI according to TS 33.223 [23].

The BSF shall check if this NAF-Id is allowed to be used for the NAF. If the NAF identified by its Origin-Host AVP is configured in the BSF not to be authorized to use the given NAF-Id, the BSF may raise the error situation 5402. The BSF may also be configured so that a certain NAF is not authorized to use a certain GAA-Service-Identifier. This situation may be also indicated by error code 5402.

- Step 3

After that the BSF shall send a GBA-Push-Info-Answer message back to the NAF. The BSF deletes the used Ks according to 3GPP TS 33.223 [23].

```
< GBA-Push-Info-Answer> ::= < Diameter Header: xxx, PXY, xxxxxxxx >
                < Session-Id >
                { Vendor-Specific-Application-Id }
                [ Result-Code ]
                [ Experimental-Result]
                { Origin-Host }                    ; Address of BSF
                { Origin-Realm }                   ; Realm of BSF
                [ User-Name ]                      ; IMPI
                [ ME-Key-Material ]                ; Required
                [ UICC-Key-Material ]              ; Conditional
                [ Key-ExpiryTime ]                 ; Time of expiry
                [ BootstrapInfoCreationTime ]      ; Bootstrapinfo creation time
                [ GBA-UserSecSettings ]            ; Selected USSs
                [ GBA-Type ]                       ; GBA type used in bootstrapping
                [ GBA-Push-Info ]                  ; GBA Push Info
                *[ AVP ]
                *[ Proxy-Info ]
                *[ Route-Record ]
```

The BSF may or may not send the User-name AVP (IMPI) It is only returned if requested and public user identity was used in the GBA-Push-Info-Request message according its configuration.

The mandatory common key material with the ME (Ks_NAF or Ks_ext_NAF) is sent in the ME-Key-Material AVP. The common key material with the UICC (Ks_int_NAF) is optionally sent in the UICC-Key-Material AVP only if the "uiccType" tag in bsfInfo from the HSS is set to "GBA_U".

The Key-ExpiryTime AVP contains the expiry time of the Bootstrapping information in the BSF according its configuration. The expiry time is represented according the Diameter Time data format in seconds that have passed since 0h on January 1, 1900 UTC . If a special key lifetime value is given in the "lifeTime" tag inside the bsfInfo from the HSS in bootstraping procedure, it is used instead of the BSF default configuration value when the expiry time is calculated.

The BootstrapInfoCreationTime AVP contains the bootstrapinfo creation time, i.e., creation time of the Bootstrapping information in the BSF. The bootstrapinfo creation time is represented in seconds that have passed since January 1, 1900 00:00:00.000 UTC.

The BSF selects the appropriate User Security Settings (if any) to the GBA-UserSecSettings AVP from stored GAA-UserSecSettings in Bootstrapping information according the GBA-Service-Identifier AVPs in the request message.

The BSF shall indicate the type of used authentication in the bootstrapping procedure to the NAF in GBA-Type, if other than 3G GBA type has been performed.

The GBA-Push-Info AVP includes the GPI. The contents of GPI are defined in 3GPP TS 33.223 [23].

When the NAF receives the GBA-Push-Info-Answer message, the NAF shall check the value of the GBA-Type AVP if it is included in the message. If the NAF does not support the GBA-Type the NAF shall stop processing the message and should indicate an error via the O&M subsystem. The further procedure in the NAF when the GBA-Push-Info-Answer message is received is described in 3GPP TS 33.223 [23], 3GPP TS 24.109 [7].

5.5 Protocol Zpn between NAF and BSF based on Web Services

The procedures in the NAF and in the BSF related Web Services [13] based Zpn interface are the same as specified in clause 5.4, but instead of Diameter messages a Web Services procedures shall be used to communicate over Zpn interface. Annex X specifies the GBA Service for Web Services, i.e., it contains the Web Services Definition Language (WSDL) [14] specification for GBA Service. Below are the attributes that are defined for GBA Service request, response, and fault cases.

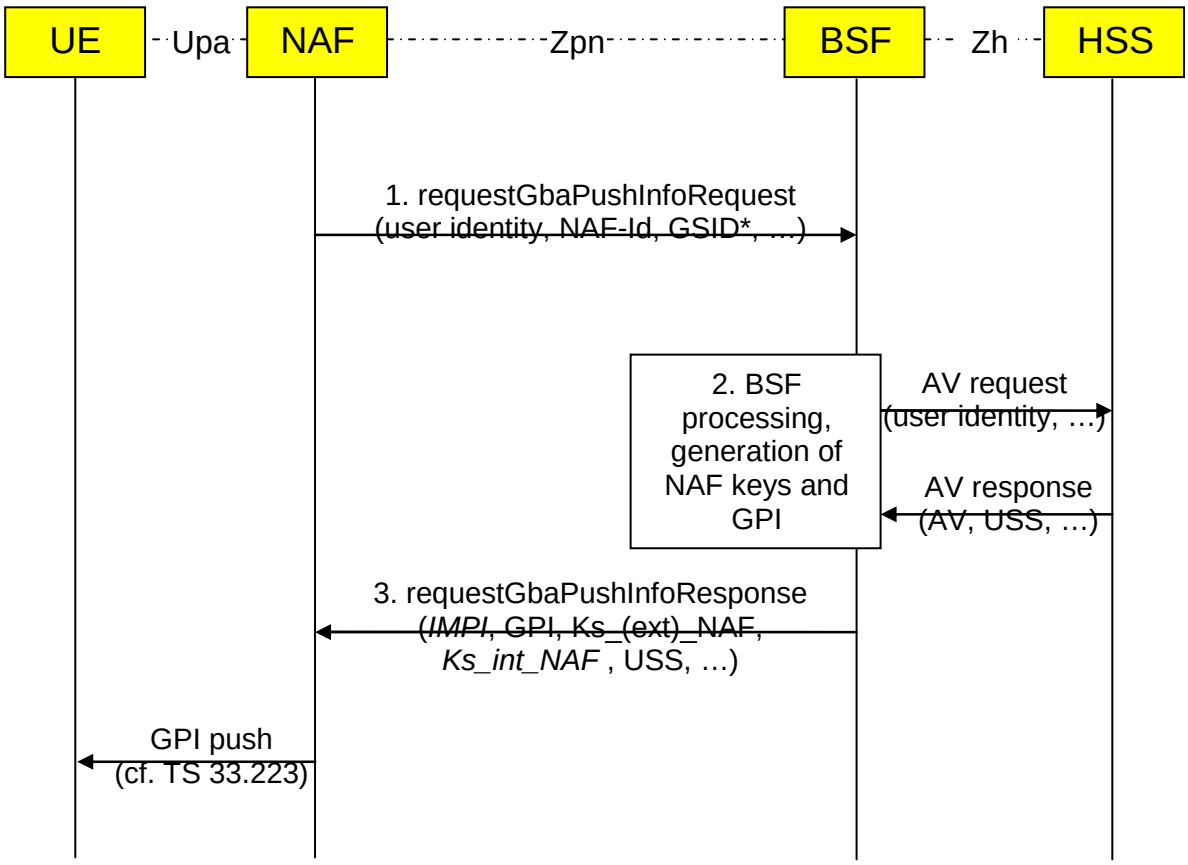


Figure 5.5: The Web Services based GAA application procedure

The possible attribute value definitions and restrictions for the request, response, and fault messages are the same as in clause 6 unless explicitly specified in this clause. The steps of the GAA application procedure in Figure 5.4 are:

Step 1

The NAF shall send a requestGbaPushInfoRequest message to the BSF. The schema of the message content is given here and it shall be in the same format as it is in WSDL.

```
<xsd:element name="requestBootstrappingInfoRequest">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ptid" type="xsd:string"/>
      <xsd:element name="nafid" type="xsd:base64Binary"/>
      <xsd:element name="gsid" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="gbaUAware" type="xsd:boolean" minOccurs="0"/>
      <xsd:element name="userId" type="xsd:string"/>
      <xsd:element name="userIdType" type="xsd:boolean" minOccurs="0"/>
      <xsd:element name="uiccAppLabel" type="xsd:string"/>
      <xsd:element name="uiccOrMe" type="xsd:boolean" minOccurs="0"/>
      <xsd:element name="requestedLifeTime" type="xsd:dateTime"/>
      <xsd:element name="privateIdRequest" type="xsd:boolean" minOccurs="0"/>
      <xsd:element name="extension" type=" typens:tExtension" minOccurs="0"/>
```

```
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

The SOAP message that shall be sent to BSF, i.e., the URI of the message shall contain the BSF address.

In the case the NAF is in a visited network, the NAF contacts the subscriber's home BSF through a GBA-Proxy that is located in the same network as the NAF. The local BSF and the GBA-Proxy may be co-located. See 3GPP TS 33.220 [6].

The NAF indicates the GAA services for which the information is retrieved by "gsid" elements. The "ptid" element is given to the BSF so that it can be included in the encrypted gbaPushInfo element. The "gbaUAware" element indicates whether NAF is GBA_U aware, and is capable of using and handling the "uiccKeyMaterial". The default value for "gbaUAware" is false. With userId and userIdType elements the NAF indicates the user identity and its type, i.e. private or public. The NAF uses uiccOrMe and uiccAppLabel elements to identity which UICC application is to be used and if GBA-ME or GBA_U should be used. privateIdRequest element indicates if he NAF requests the BSF to send the private user identity. requestedLifeTime element indicates the requested key lifetime for the NAF keys.

The NAF may use one or more "extension" elements to include additional data to the request, but the BSF may ignore the additional data.

Step 2

The procedures for step 2 are the same as in step 2 in clause 5.4.

Step 3

After that the requestGbaPushInfoResponse is sent back to the NAF.

```
<xsd:element name="requestBootstrappingInfoResponse">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="impi" type="xsd:string" minOccurs="0"/>
      <xsd:element name="meKeyMaterial" type="xsd:base64Binary"/>
      <xsd:element name="uiccKeyMaterial" type="xsd:base64Binary" minOccurs="0"/>
      <xsd:element name="keyExpiryTime" type="xsd:dateTime"/>
      <xsd:element name="bootstrappingInfoCreationTime" type="xsd:dateTime"/>
      <xsd:element name="gbaType" type="xsd:string" minOccurs="0"/>
      <xsd:element name="ussList" type="xsd:string" minOccurs="0"/>
      <xsd:element name="gbaPushInfo" type="xsd:string" minOccurs="0"/>
      <xsd:element name="extension" type=" typens:tExtension" minOccurs="0"/> </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

The BSF may or may not send the "impi" element according its configuration. It is only returned if requested and public user identity was used in requestGbaPushInfo message.

The mandatory common key material with the ME (Ks_NAF or Ks_ext_NAF) is sent in the "meKeyMaterial" element. The common key material with the UICC (Ks_int_NAF) is optionally sent in the "uiccKeyMaterial" element only if the "uiccType" tag in bsfInfo from the HSS is set to "GBA_U".

The "keyExpiryTime" element contains the expiry time of the Bootstrapping information in the BSF according its configuration. If a special key lifetime value is given in the "lifeTime" tag inside the bsfInfo from the HSS in bootstraping procedure, it shall be used instead of the BSF default configuration value when the expiry time is calculated.

The "bootstrappingInfoCreationTime" element shall contain the bootstrapinfo creation time, i.e., creation time of the Bootstrapping information in the BSF.

The BSF shall select the appropriate User Security Settings (if any) into the "ussList" element from stored GAA-UserSecSettings in Bootstrapping information according "gsid" elements in the request message. The "ussList" element contains a standalone XML document whose root element shall be "ussList" element as specified in Annex A and which contains the User Security Settings selected by the BSF.

The BSF shall indicate the type of used authentication in the bootstrapping procedure to the NAF in "gbaType" element, if other than 3G GBA type has been performed.

The gbaPushInfo element includes the GPI. The contents of GPI are defined in 3GPP TS 33.223 [23]

The BSF may use one or more "extension" elements to include additional data to the request, but the NAF may ignore the additional data.

When the NAF receives the requestBootstrappingInfoResponse message, the NAF shall check the value of the "gbaType" element if it is included in the message. If the NAF does not support the GBA type the NAF shall stop processing the message and should indicate an error via the O&M subsystem. The further procedure in the NAF when the requestBootstrappingInfoResponse message is received is described in 3GPP TS 33.220 [5], 3GPP TS 33.222 [11] and optionally in GAA service type specific TSs.

6 Diameter application for Zh and, Zn and Zpn interfaces

6.0 Introduction

The Zh, Zn and Zpn interface protocols are defined as IETF vendor specific Diameter applications, where the vendor is 3GPP. The vendor identifier assigned by IANA to 3GPP (http://www.iana.org/assignments/enterprise-numbers) is 10415.

The Diameter application identifier assigned to the Zh interface application is 16777221 (allocated by IANA).

The Diameter application identifiers assigned to the Zn and Zpn interface application are 16777220 and xxxxxxxx respectively (allocated by IANA).

6.1 Command-Code values

The Zn and Zpn interfaces assign new Command-Codes 310 and xxx.

The messages in Zh interface use the same Command-Code value 303 as Multimedia-Auth-Request/Answer messages defined in 3GPP TS 29.229 [3] for Cx interface.

6.2 Result-Code AVP values

This section defines new result code values that must be supported by all Diameter implementations that conform to this specification. When one of the result codes defined here is included in a response, it shall be inside an Experimental-Result AVP and Result-Code AVP shall be absent.

6.2.1 Success

Errors that fall within the Success category are used to inform a peer that a request has been successfully completed.

The success category result codes defined in 3GPP TS 29.229 [3] for Cx interface are useless and therefore not required in Zh, Zn and Zpn interfaces.

6.2.2 Permanent failures

Errors that fall within the Permanent Failures category are used to inform the peer that the request failed, and should not be attempted again.

The Permanent failure category result codes defined in 3GPP TS 29.229 [3] for Cx interface are useless and therefore not required in Zh , Zn and Zpn interfaces.

6.2.2.1 DIAMETER_ERROR_IMPI_UNKNOWN (5401)

A message was received by the HSS for an IMPI that is unknown.

6.2.2.2 DIAMETER_ERROR_NOT_AUTHORIZED (5402)

A message was received by the BSF which the BSF can not authorize. In this case the NAF should indicate to the UE that the service is not allowed. In case of GBA push, the NAF should not contact the UE.

6.2.2.3 DIAMETER_ERROR_TRANSACTION_IDENTIFIER_INVALID (5403)

A message was received by the BSF for an invalid (e.g. unknown or expired) Bootstrapping Transaction Identifier (B-TID). In this case the NAF should request the UE to bootstrap again.

6.2.2.4 Void

6.2.2.5 Void

6.2.2.6 Void

6.2.2.7 Void

6.3 AVPs

The AVPs defined in 3GPP TS 29.229 [3] for 3GPP IMS Cx interface Multimedia-Auth-Request/Answer messages are used as they are.

The following table describes the additional new Diameter AVPs defined for the Zh, Zn and Zpn interface protocol, their AVP Code values, types, possible flag values and whether or not the AVP may be encrypted. The Vendor-Id header of all AVPs defined in this specification shall be set to 3GPP (10415).

Table 6.1: New Diameter Multimedia Application AVPs

				AVP Flag rules				
Attribute Name	**AVP Code**	**Section defined**	**Value Type**	**Must**	**May**	**Should not**	**Must not**	**May Encr.**
GBA-UserSecSettings	400	6.3.1.1	OctedString	M, V				No
Transaction-Identifier	401	6.3.1.2	OctetString	M, V				No
NAF-Id	402	6.3.1.3	OctetString	M, V				No
GAA-Service-Identifier	403	6.3.1.4	OctedString	M, V				No
Key-ExpiryTime	404	6.3.1.5	Time	M, V				No
ME-Key-Material	405	6.3.1.6	OctedString	M, V				No
UICC-Key-Material	406	6.3.1.7	OctedString	M, V				No
GBA_U-Awareness-Indicator	407	6.3.1.8	Enumerated	M, V				No
BootstrapInfoCreationTime	408	6.3.1.9	Time	M, V				No
GUSS-Timestamp	409	6.3.1.10	Time	V			M	No
GBA-Type	410	6.3.1.11	Enumerated	M, V				No
UE-Id	411	6.3.1.12	OctetString	M, V				No
UE-Id-Type	412	6.3.1.13	Enumerated	M, V				No
UICC-App-Label	413	6.3.1.14	OctetString	M, V				No

UICC-ME	414	6.3.1.15	Enumerated	M, V				No
Requested-Key-Lifetime	415	6.3.1.16	Time	M, V				No
Private-Identity-Request	416	6.3.1.17	Enumerated	M, V				No
GBA-Push-Info	417	6.3.1.18	OctedString	M, V				No
NAF-SA-Identifier	418	6.3.1.19	OctedString	M, V				No
NOTE 1: The AVP header bit denoted as "M", indicates whether support of the AVP is required. The AVP header bit denoted as "V", indicates whether the optional Vendor-ID field is present in the AVP header.								

6.3.1 Common AVPs

6.3.1.1 GBA-UserSecSettings AVP

The GAA-UserSecSettings AVP (AVP code 400) is of type OctetString. If transmitted on the Zh interface it contains GBA user security settings (GUSS). If transmitted on the Zn interface it contains the relevant USSs only. The content of GBA-UserSecSettings AVP is a XML document whose root element shall be the "guss" element for Zh interface and the "ussList" element for the Zn interface. The XML schema is defined in annex A.

6.3.1.2 Transaction-Identifier AVP

The Transaction-Identifier AVP (AVP code 401) is of type OctetString. This AVP contains the Bootstrapping Transaction Identifier (B-TID).

6.3.1.3 NAF-Id

The NAF-Id AVP (AVP code 402) is of type OctetString. This AVP contains the full qualified domain name (FQDN) of the NAF that the UE uses concatenated with the Ua security protocol identifier as specified in 3GPP TS 33.220 [5]. The FQDN of the NAF that is part of the NAF_Id may be a different domain name that with which the BSF knows the NAF.

6.3.1.4 GAA-Service-Identifier AVP

The GAA-Service-identifier AVP (AVP code 403) is of type OctedString. This AVP informs a BSF about the support of a GAA-service by the NAF. According this AVP the BSF can select the right service"s user security settings.

For 3GPP standardized services (e.g., PKI portal), the GAA-Service-Identifier (GSID) shall be in the range 0 to 999999, and the currently standardized values for GSID shall be the GAA Service Type Code of the particular service. The GAA Service Type Codes for 3GPP standardized services are defined in Annex B.

NOTE: In the future, standardized GSID values that are different than the GAA Service Type Code may be standardised (e.g. to differentiate between the services "MBMS streaming" and "MBMS download") and then several different GSID can exist within one GAA Service Type Code.

Examples: The GSID is "1" for all PKI-portals, and "4" for all MBMS services.

6.3.1.5 Key-ExpiryTime AVP

The Key-ExpiryTime AVP (AVP code 404) is of type Time. This AVP informs the NAF about the expiry time of the key.

6.3.1.6 ME-Key-Material AVP

The required ME-Key-Material AVP (AVP code 405) is of type OctetString. The NAF is sharing this key material (Ks_NAF in the case of GBA_ME or Ks_ext_NAF in the case of GBA_U) with the Mobile Equipment (ME).

6.3.1.7 UICC-Key-Material AVP

The condition UICC-Key-Material AVP (AVP code 406) is of type OctetString. The NAF may share this key material (Ks_int_NAF in the case of GBA_U) with a security element (e.g. USIM, ISIM, etc..) in the UICC. Only some GAA applications use this conditional AVP.

6.3.1.8 GBA_U-Awareness-Indicator

The conditional GBA_U-Awareness-Indicator AVP (AVP code 407) is of type Enumerated. The following values are defined.

NO (0) The sending node is not GBA_U aware

YES(1) The sending node is GBA_U aware

The default value is 0 i.e. absence of this AVP indicates that the sending node is not GBA_U aware.

6.3.1.9 BootstrapInfoCreationTime AVP

The BootstrapInfoCreationTime AVP (AVP code 408) is of type Time. This AVP informs the NAF about the bootstrapinfo cration time of the key.

6.3.1.10 GUSS-Timestamp AVP

The GUSS-Timestamp AVP (AVP code 409) is of type Time. If transmitted this AVP informs the HSS about the timestamp of the GUSS stored in the BSF.

6.3.1. 11 GBA-Type

The GBA-Type AVP (AVP code 410) is of type Enumerated. The AVP informs the NAF about the authentication type that was used during bootstrapping procedure.

The following values are defined:

- 3G GBA (0) The 3G GBA has been performed as defined in TS 33.220 [5].
- 2G GBA (1) The 2G GBA has been performed as defined in TS 33.220 [5].

The default value is 0 i.e. the absence of this AVP indicates 3G GBA

6.3.1. 12 UE-Id

The UE-Id AVP (AVP code 411) is of type OctedString. The AVP informs the BSF the identity of the user.

6.3.1. 13 UE-Id-Type

The UE-Id-Type AVP (AVP code 412) is of type Enumerated. The AVP informs the BSF the type of the identity of the user.

The following values are defined:

- (0) Private user identity.
- (1) Public user identity.

6.3.1. 14 UICC-App-Label

The UICC-App-Label AVP (AVP code 413) is of type OctedString. The AVP informs the BSF the UICC application to be used for GBA push.

6.3.1. 15 UICC-ME

The UICC-ME AVP (AVP code 414) is of type Enumerated. The AVP informs the BSF the whether GBA_ME or GBA_U is to be used for GBA push.

The following values are defined:

- GBA_ME (0) GBA_ME shall be run.
- GBA_U (1) GBA_U shall be run.

6.3.1. 16 Requested-Key-Lifetime

The Requested-Key-Lifetime AVP (AVP code 415) is of type Time. The AVP informs the BSF about the requested lifetime for the NAF keys.

6.3.1. 17 Private-Identity-Request

The Private-Identity-Request AVP (AVP code 416) is of type Enumerated. The AVP informs the BSF if the NAF requests the private identity of the user.

The following values are defined:

- Private identity requested (0).
- Private identity not requested (1)

6.3.1. 18 GBA-Push-Info

The GBA-Push-Info AVP (AVP code 417) is of type OctetString. The AVP includes the GBA-Push-Info as defined in 3GPP TS 33.223 [23].

6.3.1. 19 NAF-SA-Identifier

The NAF-SA-Identifier AVP (AVP code 418) is of type OctetString. The AVP contains the NAF-SA-Identifier (P-TID). See 3GPP TS 33.223 [23].

6.4 User identity to HSS resolution

The User identity to HSS resolution mechanism enables the BSF to find the identity of the HSS that holds the subscriber data for a given subscriber when multiple and separately addressable HSSs have been deployed by the network operator. The resolution mechanism is not required in networks that utilise a single HSS or when a BSF is configured to use pre-defined HSS.

The resolution mechanism is equivalent to the mechanism defined for the Cx/Dx interface described in 3GPP TS 23.228 [12] and shall use a Subscription Locator Function (SLF) or a Diameter Proxy Agent.

The BSF accesses the SLF via the Dz interface. The Dz interface shall be always used in conjunction with the Zh interface. The Dz interface is based on Diameter. The SLF functionality shall use the routing mechanism provided by an enhanced Diameter redirect agent, which is able to extract the Subscriber identity from the received requests.

The SLF or the Diameter Proxy Agent shall be able to determine the HSS identity.

To get the HSS identity the BSF shall send the Zh request normally destined to the HSS to a pre-configured Diameter identity.

- If this Zh request is received by an SLF (acting as a Diameter redirect agent), the SLF shall determines the HSS identity and shall send to the BSF a notification of redirection towards the HSS identity, in response to the Zh request. Multiple HSS identities may be included in the response, as specified in IETF RFC 3588 [1]. In such a case, the BSF shall send the Zh Request to the first HSS identity in the ordered list received in the Zh Response from the SLF. If the BSF does not receive a successful response to the Zh Request, the BSF shall send a Zh

Request to the next HSS identity in the ordered list. This procedure shall be repeated until a successful response from an HSS is received.

- If this Zh request is received by the Diameter Proxy Agent, the Diameter Proxy Agent shall determine the HSS identity and shall forward the Zh request directly to the HSS. The BSF shall determine the HSS identity from the response to the Sh request received from the HSS.

After the user identity to HSS resolution, the BSF may store the HSS identity and, if stored, shall use it in further Zh requests associated to the same Subscriber.

In networks where the use of the user identity to HSS resolution mechanism is required and the BSF is not configured to use predefined HSS, each BSF shall be configured with the address/name of the SLF or the Diameter Proxy Agent implementing this resolution mechanism.

7 Use of namespaces

This clause contains the namespaces that have either been created in this 3GPP specification, or in 3GPP specification 3GPP TS 29.229 [3] or the values assigned to existing namespaces managed by IANA.

7.1 AVP codes

This specification reserves the 3GPP vendor specific values 10415:400-499 and assigns values 10415:400-410 for the GAA from the 3GPP AVP Code namespace for 3GPP Diameter applications ([8]). The 3GPP vendor specific AVP code space is managed by 3GPP CT4. See section 6 for the assignment of the namespace in this specification.

Besides the Diameter Base Protocol AVPs [1] this specification reuses the following AVPs from 3GPP TS 29.229 [3]: `Authentication-Session-State, User-Name, Public-User-Identity` and `SIP-Auth-Data-Item`.

7.2 Experimental-Result-Code AVP values

This specification reserves Experimental-Result-Code AVP values 10415:2401-2409 and 10415:5401-5409. See section 6.2.

7.3 Command Code values

Only Command-Codes 310 and 303 from 3GPP TS 29.229 [3] is used in this specification.

This specification reuses only the Command-Code value, not the content of the original specification. The AVPs, that are defined required in TS 29.229 [3], but are not needed in Zh or Zn interfaces, are removed and are therefore not required in Zh or Zn interface messages. All new AVPs for GAA are defined optional although they may be mandatory in GAA viewpoint.

This specification does not assign new command codes to the 3GPP TS 29.229 [3].

Annex A (normative): GBA-UserSecSettings XML definition

This annex contains the XML schema definition for an XML document carrying the GBA User Security Settings inside GBA-UserSecSettings AVP in Zh and Zn interface.

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace="urn:3gpp:gba:GBAGUSSSchema-R7:2008-01"
    xmlns:tns="urn:3gpp:gba:GBAGUSSSchema-R7:2008-01"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <!-- This import brings in the XML language attribute xml:lang-->
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
      schemaLocation="http://www.w3.org/2001/xml.xsd"/>

  <!-- Root element to be used in Zh reference point-->
  <xs:element name="guss" type="tns:gussType"/>

  <!-- Root element to be used in Zn reference point-->
  <xs:element name="ussList" type="tns:ussListType"/>

  <xs:complexType name="ExtensionType">
    <xs:sequence>
      <xs:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="GUSSExtensionType">
    <xs:sequence>
      <xs:element name="timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="Extension" type="tns:ExtensionType" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="USSExtensionType">
    <xs:sequence>
      <xs:element name="keyChoice" type="xs:string" minOccurs="0" />
      <xs:element name="Extension" type="tns:ExtensionType" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <!-- The whole user"s GBA specific data set  -->
  <xs:complexType name="gussType">
    <xs:sequence>
       <xs:element name="bsfInfo" type="tns:bsfInfoType" minOccurs="0"/>
       <xs:element name="ussList" type="tns:ussListType"/>
       <xs:element name="Extension" type="tns:GUSSExtensionType" minOccurs="0"/>
       <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:string"/>
  </xs:complexType>

  <!-- BSF specific information element -->
  <xs:complexType name="bsfInfoType">
    <xs:sequence>
      <xs:element name="uiccType"  type="xs:string"  minOccurs="0" />
```

```
      <xs:element name="lifeTime"  type="xs:integer" minOccurs="0" />
      <xs:element name="Extension" type="tns:ExtensionType" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!--List of all users individual User Security Settings -->
  <xs:complexType name="ussListType">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
       <xs:element name="uss" type="tns:ussType"/>
       <xs:element name="Extension" type="tns:ExtensionType" minOccurs="0"/>
       <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- User Security Setting data -->
  <xs:complexType name="ussType">
    <xs:sequence>
      <xs:element name="uids" type="tns:uidsType"/>
      <xs:element name="flags" type="tns:flagsType"/>
      <xs:element name="Extension" type="tns:USSExtensionType" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id"   use="required" type="xs:string"/>
    <xs:attribute name="type" use="required" type="xs:int"/>
    <xs:attribute name="nafGroup" use="optional" type="xs:string"/>
  </xs:complexType>

  <!-- User Public Identities for authentication -->
  <xs:complexType name="uidsType">
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
      <xs:element name="uid"  type="xs:string"/>
      <xs:element name="Extension" type="tns:ExtensionType" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- GAA Application type specific Authorization flag codes -->
  <xs:complexType name="flagsType">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="flag"  type="xs:int"/>
      <xs:element name="Extension" type="tns:ExtensionType" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

NOTE 1: The <xs:any> elements within the complex types ExtensionType allow for compatible standard extensions in future releases. The <xs:any namespace=##other"> elements within the other complex types allow for compatible private extensions.

The values are:

- The value of the attribute "id" in the element "guss" is the same as user"s IM Private Identity (IMPI) used in User-Name AVP.

- The value of the element "timestamp" in the element "guss" is the same type as GUSS-Timstamp used in GUSS-Timestamp AVP and indicates the timestamp of the GUSS. Timestamp value shall be expressed in UTC form, indicated by a time zone designator "Z" immediately following the time portion of the value.

- The value of the attribute "id" in the element "uss" is the same as service identifier (GSID) used in GAA-Service-Identifier AVP.

NOTE 2: In the case with currently standardized 3GPP applications (c. f. Annex B), the service identifier (GSID) is the same as the GAA Service Type Code i.e. the "id" and the "type" attribute would have the same value. For example, in the interoperator GAA where the requesting BM-SC (i.e. NAF) is different operator network than the answering BSF, the BM-SC (NAF) can request particular user's MBMS USS by using "4" for the GSID in the "id" attribute in the USS. If the BSF operator wishes to have different MBMS USSs for different BM-SCs (NAFs), it can use the nafGroup attribute to separate NAFs to specific groups, and each group would get a particular USSs: <uss id="4" type="4" nafGroup="A"> would be given to NAFs in group A, and <uss id="4" type="4" nafGroup="B"> would be given to NAFs in group B when they request it. NAF groups are operator specific, i.e., operator decides which USS is given to which NAF."

- The value of the element "uiccType" in the element "bsfInfo" is:
  GBA to indicate the basic case, or
  GBA_U to indicate that generation of Ks_int_NAF is also required in the BSF.
  The default value is GBA.

- The value of the element "lifeTime" in the element "bsfInfo" indicates a user specific key lifetime (duration in seconds). If the lifeTime element is missing the default value in the BSF is used.

- The value of attribute "type" in the element "uss" is GAA service type code defined in annex B.

- The value of attribute "nafGroup" in the element "uss" is an operator internal group designator for a NAF group the USS is valid for. If this attribute is missing then only the attribute "id" is used for selection of this element.

- Values of the element "uid" are user"s public authentication identities from the HSS.

- Values of element "flag" are user"s authorization flag codes from the HSS for GAA service type indicated in the type attribute in the parent uss element. If an authorization flag exist the NAF have permission to give the corresponding service, otherwise not

- The value of the element "keyChoice" in the "Extension" element inside the "uss" element is "ME-based-key", i.e., Ks_ NAF or Ks_ext_NAF shall be used, or "UICC-based-key", i.e., Ks_int_NAF shall be used or "ME-UICC-based-keys", i.e., Ks_ext_NAF or Ks_int_NAF can be used. The value of this element indicates to the NAF, which key shall be used. If the keyChoice element is missing, then as a default the "ME-based-key" shall be used by the NAF unless a prior agreement exists that mandates to use "UICC-based-key" or "ME-UICC-based-keys".

In the following illustrative example the values are italised and underlined. The content of one User Security Setting tag is boxed.

```
<guss id="358500004836551@ims.mnc050.mcc358.3gppnetwork.org">
     <bsfInfo>
          <lifeTime>86400</lifeTime>
     </bsfInfo>
     <ussList>
          <uss id="1" type="1" nafGroup="A">
               <uids>
                    <uid>tel:358504836551</uid>
                    <uid>lauri.laitinen@example.com</uid>
                    …
               </uids>
               <flags>
                    <flag>1</flag>
                    …
               </flags>
               <Extension>
                    <keyChoice>ME-based-key</keyChoice>
               </Extension>
```

```
        </uss>
        …
    </ussList>
</guss>
```

The above GAA User Security Settings example for user "358500004836551@ ims.mnc050.mcc358.3gppnetwork.org" defines that for PKI-Portal (GAA service type code is 1) services are allowed for user identities "tel:358504836551" and "lauri.laitinen@example.com" and authentication is allowed (flag 1 exists) but non-repudiation is not allowed (flag 2 is missing) to NAFs that provide the GAA service identified by "1" GAA Service Identifier. This particular USS for PKI-Portal is intended to be used only with a NAF group that is labeled as "A" (NAF groupings and NAFs that belong to these groups are specified by the MNO). Additionally, the key choice for the PKI-Portal should use only ME based key (as key keyChoice is set to "ME-based-key"). The BSF shall not generate UICC-Ks, because uiccType is missing. A special key lifetime defines that a the duration after which the key expires is 86400 seconds.

```
<![CDATA[<?xml version='1.0'?>
<ussList>
     <uss id="1" type="1">
          <uids>
               <uid>tel:358504836551</uid>
               <uid>lauri.laitinen@example.com</uid>
          </uids>
          <flags>
               <flag>1</flag>
          </flags>
          <Extension>
               <keyChoice>ME-based-key</keyChoice>
          </Extension>
     </uss>
</ussList>
]]>
```

The above is an example how the value of "GAA-UserSecSettings" in Diameter based Zn reference point and "ussList" in Web Services based Zn reference point is populated.

NOTE: The BSF has removed the "nafGroup" attribute from "uss" element. Also, the XML document has been surrounded by "<![CDATA[" and "]]>" tags, so that the XML parser handling the outer message (i.e., in Web Services case) will not parse the ussList. The ussList will be parsed by the application itself handling the incoming message.

Annex B (normative): GAA Service Type Codes

The GAA Service Type Code values are used in GAA to indicate interpretation, coding and usage of GAA service type specific data.

For examples each GAA service type may have their own set of authorization flags. Meaning and coding of these flags are defined in Annex C. There may also be proprietary GAA service types with their own definitions in the future.

Code values 0 – 999999 are reserved for standardized GAA service types.

The following values are defined for standardized GAA service types with 3GPP specification:

0 Unspecific service
1 PKI-Portal
2 Authentication Proxy
3 Presence
4 MBMS
5 Liberty Alliance Project (see [15])
6 UICC - Terminal Key Establishment (see [17])
7 Terminal – Remote Device Key Establishment (see [18])

Default value is 0. An unspecific service may or may not have user security settings containing or not a list of public identities. An unspecific service cannot have specified authorization flags or other service type specific data.

Annex C (normative): GAA Authorization flag codes

For GAA services which have a defined set of special authorization flag codes the following rule holds: The service specified by the GAA authorization flag codes is allowed for a user only if user"s user security setting contains that flag.

The following standardised GAA service types that are listed in previous annex B have the following special authorization flag codes:

PKI-Portal (1)

1 Authentication allowed

2 Non-repudiation allowed

Annex D (normative): Web Services Definition for Zn interface

This annex contains the Web Services Defination Language (WSDL) [14] for Zn interface:

```
<?xml version="1.0" encoding="UTF-8"?>

<wsdl:definitions name="GBAService"
                  targetNamespace="urn:3gpp:gba:GBAService:2007-05"
                  xmlns:typens="urn:3gpp:gba:GBAService:2007-05"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">

  <wsdl:types>

    <xsd:schema targetNamespace="urn:3gpp:gba:GBAService:2007-05">

      <!-- Extension element definition -->
      <xsd:complexType name="tExtension">
        <xsd:sequence>
          <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:complexType>

      <!-- Request Bootstrapping info request parameter definitions -->
      <xsd:element name="requestBootstrappingInfoRequest">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="btid" type="xsd:string"/>
            <xsd:element name="nafid" type="xsd:base64Binary"/>
            <xsd:element name="gsid" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="gbaUAware" type="xsd:boolean" minOccurs="0"/>
            <xsd:element name="extension" type="typens:tExtension" minOccurs="0"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>

      <!-- Request Bootstrapping info responset parameter definitions -->
      <xsd:element name="requestBootstrappingInfoResponse">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="impi" type="xsd:string" minOccurs="0"/>
            <xsd:element name="meKeyMaterial" type="xsd:base64Binary"/>
            <xsd:element name="uiccKeyMaterial" type="xsd:base64Binary" minOccurs="0"/>
            <xsd:element name="keyExpiryTime" type="xsd:dateTime"/>
            <xsd:element name="bootstrappingInfoCreationTime" type="xsd:dateTime"/>
            <xsd:element name="gbaType" type="xsd:string" minOccurs="0"/>
            <xsd:element name="ussList" type="xsd:string" minOccurs="0"/>
            <xsd:element name="extension" type="typens:tExtension" minOccurs="0"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>

      <!-- Request Bootstrapping info fault parameter definitions -->
      <xsd:element name="requestBootstrappingInfoFault">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="errorCode" type="xsd:integer"/>
            <xsd:element name="errorText" type="xsd:string" minOccurs="0"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>

    </xsd:schema>

  </wsdl:types>

  <wsdl:message name="requestBootstrappingInfoRequestMessage">
    <wsdl:part name="body" element="typens:requestBootstrappingInfoRequest"/>
  </wsdl:message>
  <wsdl:message name="requestBootstrappingInfoResponseMessage">
    <wsdl:part name="body" element="typens:requestBootstrappingInfoResponse"/>
```

```
  </wsdl:message>
  <wsdl:message name="requestBootstrappingInfoFaultMessage">
    <wsdl:part name="body" element="typens:requestBootstrappingInfoFault"/>
  </wsdl:message>

  <wsdl:portType name="GBAServicePortType">
    <wsdl:operation name="requestBootstrappingInfo">
      <wsdl:input message="typens:requestBootstrappingInfoRequestMessage"/>
      <wsdl:output message="typens:requestBootstrappingInfoResponseMessage"/>
      <wsdl:fault name="FaultName" message="typens:requestBootstrappingInfoFaultMessage"/>
    </wsdl:operation>
  </wsdl:portType>

  <wsdl:binding name="GBAServiceBinding" type="typens:GBAServicePortType">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="requestBootstrappingInfo">
      <soap:operation soapAction="urn:3gpp:gba:GBAServiceAction:2007-05"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
      <wsdl:fault name="FaultName">
        <soap:fault name="FaultName" use="literal"/>
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>

  <wsdl:service name="GBAService">
    <wsdl:port name="GBAServicePort" binding="typens:GBAServiceBinding">
      <!-- add SOAP address location URI below -->
      <soap:address location="http://add.here.uri.to/GBAService"/>
    </wsdl:port>
  </wsdl:service>

</wsdl:definitions>
```

Annex E (informative): Liberty authentication context definitions for GBA

E.1 Introduction

This clause describes the GBA Authentication Context definition that the Liberty Identity Provider uses to describe GBA parameters to Liberty Service Provider. The Liberty 3GPP Security Interworking is further discussed in 3GPP TR 33.980 [15].

E.2 GBA Authentication context statement data model

A particular Liberty authentication context statement will capture the characteristics of the process, procedures, and mechanisms by which the authentication authority verified the subject before issuing an identity, protects the secrets on which subsequent authentications are based, and the mechanisms used for this authentication. These characteristics are categorized in the Liberty ID-FF Authentication Context Specification [16] as follows:

- Identification - Characteristics that describe the processes and mechanism the authentication authority uses to initially create an association between a subject and the identity (or name) by which the subject will be known.
- Technical Protection - Characteristics that describe how the "secret" (the knowledge or possession of which allows the subject to authenticate to the authentication authority) is kept secure.
- Operational Protection - Characteristics that describe procedural security controls employed by the authentication authority (for example, security audits, records archival).
- Authentication Method - Characteristics that define the mechanisms by which the subject of the issued assertion authenticates to the authentication authority (for example, a password versus a smartcard).
- Governing Agreements - Characteristics that describe the legal framework (e.g. liability constraints and contractual obligations) underlying the authentication event and/or its associated technical authentication infrastructure.

Compared to Liberty Authentication Context [16], the GBA Authentication Context statement data model adds a description to to which GBA mechanism was used during bootstrapping with the BSF, and how the corresponding shared secret was used with the Identity Provider (IdP) that functions as a NAF.

The authentication contexts classes specific to mobile terminals specified by Liberty Alliance are called

- MobileOneFactorUnregistered,
- MobileTwoFactorUnregistered,
- MobileOneFactorContract, and
- MobileTwoFactorContract classes.

The GBA authentication context uses some of the elements in these classes and introduces a new element called "GBAMechanism", which can be used to describe GBA specific parameters of the authentication. The elements used in the GBA Authentication Context schemas are:

- GBAMechanismType: describes the GBA bootstrapping mechanism (identifies the method used over Ub reference point);
- AuthenticatorTransportProtocolType: describes the authentication mechanism that was used authenticate the UE towards the Liberty Identity Provider (identifies the method used over Ua reference point);
- KeyActivationType: describes the mechanism to achieve two factor authentication (valid in *TwoFactor* authentication schemas).

E.3 GBA authentication context statement schema

This section lists the complete GBA Authentication Context XML schema. It is based on Liberty Authentication Context XML Schema to which the addition is the additional description of how GBA procedures have been conducted, i.e., GBAMechanism element.

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace="urn:3gpp:gba:ac:2006-10"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ac="urn:liberty:ac:2004-12"
  xmlns="urn:3gpp:gba:ac:2006-10">

  <!-- imports Liberty Authentication Context definitions -->
  <xs: import namespace="urn:liberty:ac:2004-12"
    schemaLocation="liberty-authentication-context-v2.0.xsd"/>

  <xs:annotation>
    <xs:documentation>
      This authentication context has been defined for the
      3GPP and 3GPP2 Generic Bootstrapping Architecture. It
      defines new GBAMechanismType and its values, but reuses
      the Liberty authentication context schema for other
      values. The GBA authentication context is based on the
      MobileOneFactor* and MobileTwoFactor* authentication
      contexts with the exception that only shared secret
      based authentication methods (i.e., symmetric) are used,
      and private key (i.e., asymmetric) methods are not
      used when authenticating the UE due to the nature of GBA.
    </xs:documentation>
  </xs:annotation>

  <!-- new type definitions for different GBA procedures -->
  <xs:element name="LegacyGBA">
    <xs:annotation>
      <xs:documentation>
        Legacy GBA where existing old authentication
        frameworks are used for bootstrapping such
        2G GBA in 3GPP, and CDMA 1x and CDMA 1xEvDo
        in 3GPP2.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:element name="GBAMobileMobile">
    <xs:annotation>
      <xs:documentation>
        GBA using AKA as specified in 3GPP and 3GPP2.
        The shared secret is derived in the mobile and
        used in the mobile.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:element name="GBAUICCMobile">
    <xs:annotation>
      <xs:documentation>
        GBA_U using AKA as specified in 3GPP and 3GPP2.
        The shared secret is derived in the UICC and used
        in the mobile.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:element name="GBAUICCUICC">
    <xs:annotation>
      <xs:documentation>
        GBA_U using AKA as specified in 3GPP and 3GPP2.
        The shared secret is derived in the UICC and used
        in the UICC.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:element name="GBAMechanism" type="GBAMechanismType">
    <xs:annotation>
```

```
      <xs:documentation>
        GBA mechanism used in the bootstrapping procedure.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:complexType name="GBAMechanismType">
    <xs:choice>
      <xs:element ref="LegacyGBA"/>
      <xs:element ref="GBAMobileMobile"/>
      <xs:element ref="GBAUICCMobile"/>
      <xs:element ref="GBAUICCUICC"/>
    </xs:choice>
  </xs:complexType>

</xs:schema>
```

E.4 GBA authentication context classes

E.4.1 GBAOneFactorUnregistered

This class reflects that there were no mobile customer registration procedures and an authentication of the UE is done without requiring explict end-user interaction.

E.4.1.1 Associated 3GPP URI

http://www.3gpp.org/schemas/authctx/classes/GBAOneFactorUnregistered

E.4.1.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace="urn:3gpp:gba:ac:2006-10"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ac="urn:liberty:ac:2004-12"
  xmlns="urn:3gpp:gba:ac:2006-10"
  finalDefault="extension"
  version="1.0">

  <!-- Imports Liberty authentication context schema definitions -->
  <xs:import namespace="urn:liberty:ac:2004-12"
    schemaLocation="liberty-authentication-context-v2.0.xsd"/>

  <!-- includes 3GPP GBA Authentication Context generic definitions -->
  <xs:include schemaLocation="3gpp-gba-authentication-context-v1.0.xsd"/>

  <!-- GBAOneFactorUnregistered -->

  <xs:complexType name="GBAOneFactorUnregisteredGBAMechanismType">
    <xs:complexContent>
      <xs:restriction base="GBAMechanismType">
        <xs:choice>
          <xs:element ref="LegacyGBA"/>
          <xs:element ref="GBAMobileMobile"/>
          <xs:element ref="GBAUICCMobile"/>
          <xs:element ref="GBAUICCUICC"/>
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <!-- states which protocol was used between UE and NAF/IdP to authenticate the UE -->
  <xs:complexType name="GBAOneFactorUnregisteredAuthenticatorTransportProtocolType">
    <xs:complexContent>
      <xs:restriction base="ac:AuthenticatorTransportProtocolType">
        <xs:choice>
          <xs:element ref="ac:HTTP"/> <!-- HTTP Digest over SSL/TLS -->
          <xs:element ref="ac:SSL"/>  <!-- PSK TLS -->
```

```
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

</xs:schema>
```

E.4.2 GBATwoFactorUnregistered

This class reflects that there were no mobile customer registration procedures and a two-factor based authentication of the UE is done requiring an explicit end-user interaction during authentication procedure (e.g., a PIN needs to be typed).

E.4.2.1 Associated 3GPP URI

http://www.3gpp.org/schemas/authctx/classes/GBATwoFactorUnregistered

E.4.2.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace="urn:3gpp:gba:ac:2006-10"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ac="urn:liberty:ac:2004-12"
  xmlns="urn:3gpp:gba:ac:2006-10"
  finalDefault="extension"
  version="1.0">

  <!-- Imports Liberty authentication context schema definitions -->
  <xs:import namespace="urn:liberty:ac:2004-12"
    schemaLocation="liberty-authentication-context-v2.0.xsd"/>

  <!-- includes 3GPP GBA Authentication Context generic definitions -->
  <xs:include schemaLocation="3gpp-gba-authentication-context-v1.0.xsd"/>

  <!-- GBATwoFactorUnregistered -->

  <xs:complexType name="GBATwoFactorUnregisteredGBAMechanismType">
    <xs:complexContent>
      <xs:restriction base="GBAMechanismType">
        <xs:choice>
          <xs:element ref="LegacyGBA"/>
          <xs:element ref="GBAMobileMobile"/>
          <xs:element ref="GBAUICCMobile"/>
          <xs:element ref="GBAUICCUICC"/>
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <!-- states which protocol was used between UE and NAF/IdP to authenticate the UE -->
  <xs:complexType name="GBATwoFactorUnregisteredAuthenticatorTransportProtocolType">
    <xs:complexContent>
      <xs:restriction base="ac:AuthenticatorTransportProtocolType">
        <xs:choice>
          <xs:element ref="ac:HTTP"/> <!-- HTTP Digest over SSL/TLS -->
          <xs:element ref="ac:SSL"/>  <!-- PSK TLS -->
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="GBATwoFactorUnregisteredKeyActivationType">
    <xs:complexContent>
      <xs:restriction base="ac:KeyActivationType">
        <xs:choice>
          <xs:element ref="ac:ActivationPin"/>
          <xs:element ref="ac:Extension" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
```

```
</xs:schema>
```

E.4.3 GBAOneFactorContract

This class reflects that mobile customer registration procedures have taken place and an authentication of the UE is done without requiring explicit end-user interaction.

E.4.3.1 Associated 3GPP URI

http://www.3gpp.org/schemas/authctx/classes/GBAOneFactorContract

E.4.3.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace="urn:3gpp:gba:ac:2006-10"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ac="urn:liberty:ac:2004-12"
  xmlns="urn:3gpp:gba:ac:2006-10"
  finalDefault="extension"
  version="1.0">

  <!-- Imports Liberty authentication context schema definitions -->
  <xs:import namespace="urn:liberty:ac:2004-12"
    schemaLocation="liberty-authentication-context-v2.0.xsd"/>

  <!-- includes 3GPP GBA Authentication Context generic definitions -->
  <xs:include schemaLocation="3gpp-gba-authentication-context-v1.0.xsd"/>

  <!-- GBAOneFactorContract -->

  <xs:complexType name="GBAOneFactorContractGBAMechanismType">
    <xs:complexContent>
      <xs:restriction base="GBAMechanismType">
        <xs:choice>
          <xs:element ref="LegacyGBA"/>
          <xs:element ref="GBAMobileMobile"/>
          <xs:element ref="GBAUICCMobile"/>
          <xs:element ref="GBAUICCUICC"/>
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <!-- states which protocol was used between UE and NAF/IdP to authenticate the UE -->
  <xs:complexType name="GBAOneFactorContractAuthenticatorTransportProtocolType">
    <xs:complexContent>
      <xs:restriction base="ac:AuthenticatorTransportProtocolType">
        <xs:choice>
          <xs:element ref="ac:HTTP"/> <!-- HTTP Digest over SSL/TLS -->
          <xs:element ref="ac:SSL"/>  <!-- PSK TLS -->
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="GBAOneFactorContractIdentificationType">
    <xs:complexContent>
      <xs:restriction base="ac:IdentificationType">
        <xs:sequence>
          <xs:element ref="ac:PhysicalVerification"/>
          <xs:element ref="ac:WrittenConsent"/>
          <xs:element ref="ac:Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

</xs:schema>
```

E.4.4 GBATwoFactorContract

This class reflects that mobile customer registration procedures have taken place and a two-factor based authentication of the UE is done requiring an explicit end-user interaction during authentication procedure (e.g., a PIN needs to be typed).

E.4.4.1 Associated 3GPP URI

http://www.3gpp.org/schemas/authctx/classes/GBATwoFactorContact

E.4.4.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace="urn:3gpp:gba:ac:2006-10"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ac="urn:liberty:ac:2004-12"
  xmlns="urn:3gpp:gba:ac:2006-10"
  finalDefault="extension"
  version="1.0">

  <!-- Imports Liberty authentication context schema definitions -->
  <xs:import namespace="urn:liberty:ac:2004-12"
    schemaLocation="liberty-authentication-context-v2.0.xsd"/>

  <!-- includes 3GPP GBA Authentication Context generic definitions -->
  <xs:include schemaLocation="3gpp-gba-authentication-context-v1.0.xsd"/>

  <!-- GBATwoFactorContract -->

  <xs:complexType name="GBATwoFactorContractGBAMechanismType">
    <xs:complexContent>
      <xs:restriction base="GBAMechanismType">
        <xs:choice>
          <xs:element ref="LegacyGBA"/>
          <xs:element ref="GBAMobileMobile"/>
          <xs:element ref="GBAUICCMobile"/>
          <xs:element ref="GBAUICCUICC"/>
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <!-- states which protocol was used between UE and NAF/IdP to authenticate the UE -->
  <xs:complexType name="GBATwoFactorContractAuthenticatorTransportProtocolType">
    <xs:complexContent>
      <xs:restriction base="ac:AuthenticatorTransportProtocolType">
        <xs:choice>
          <xs:element ref="ac:HTTP"/> <!-- HTTP Digest over SSL/TLS -->
          <xs:element ref="ac:SSL"/>  <!-- PSK TLS -->
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="GBATwoFactorContractKeyActivationType">
    <xs:complexContent>
      <xs:restriction base="ac:KeyActivationType">
        <xs:choice>
          <xs:element ref="ac:ActivationPin"/>
          <xs:element ref="ac:Extension" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="GBATwoFactorContractIdentificationType">
    <xs:complexContent>
      <xs:restriction base="ac:IdentificationType">
        <xs:sequence>
```

```
          <xs:element ref="ac:PhysicalVerification"/>
          <xs:element ref="ac:WrittenConsent"/>
          <xs:element ref="ac:Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

</xs:schema>
```

Annex F (informative): SAML authentication context definitions for GBA

F.1 Introduction

This clause describes the GBA Authentication Context definition that an Identity Provider (IdP) uses to describe GBA parameters to a Service Provider (SP). SAML authentication contexts are further discussed in the SAML Authentication Context specification [19].

F.2 GBA authentication context declaration data model

Similarly to Liberty (cf. Annex E), a SAML authentication context declaration captures characteristics of the processes, procedures, and mechanisms by which the authentication authority verified the subject before issuing an identity, protects the secrets on which subsequent authentications are based, and the mechanisms used for this authentication (identification, technical protection, operational protection, authentication method and governing agreements) OASIS Standard [22].

Compared to the SAML authentication context declaration data model OASIS Standard [22], the GBA authentication context declaration data model extends the SAML model by adding a description of which GBA mechanism was used during bootstrapping with the BSF, and at the same time restricts the SAML model in a number of respects specific to GBA (only shared secret as authenticator type, only HTTP or SSL as authenticator transport protocol, secret key activation by PIN, only mobile device or smartcard as key storage medium).

The authentication contexts classes specific to mobile terminals specified by SAML are:

- MobileOneFactorUnregistered,
- MobileTwoFactorUnregistered,
- MobileOneFactorContract, and
- MobileTwoFactorContract.

The GBA authentication context classes introduced herein correspond to the above ones and are:

- GBAOneFactorUnregistered,
- GBATwoFactorUnregistered,
- GBAOneFactorContract, and
- GBATwoFactorContract.

The GBA authentication context classes are derived from the corresponding SAML authentication context classes by means of the following patterns:

- The GBA mechanism (corresponding to ME-based GBA, UICC-based GBA, 2G GBA, etc.) is added to the authentication method type for all the four classes.
- A set of GBA-specific restrictions (only shared secret as authenticator type, only HTTP or SSL as authenticator transport protocol, only mobile device or smartcard as key storage medium) is applied to all the four classes.
- For the 'OneFactor' classes (GBAOneFactorUnregistered and GBAOneFactorContract), there is no more difference (compared to the corresponding SAML class) in addition to the two changes described above.
- For the 'TwoFactor' classes (GBATwoFactorUnregistered and GBATwoFactorContract), the key activation by means of PIN is mandatory in secret key protection.

Given the above patterns, the rest of this annex is expected to be understood without further detailed explanations.

F.3 GBA authentication context declaration types

This section lists the type definitions common to all the four GBA authentication context declaration classes. It adds the GBA mechanism elements (and the necessary type definitions) to the SAML types.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:documentation>
      These types are intended to be used in the definition of
      the authentication context classes in 3GPP and 3GPP2 Generic
      Bootstrapping Architecture (GBA). The new authentication
      context classes are defined separately.
    </xs:documentation>
  </xs:annotation>
  <!-- SAML authentication context types -->
  <xs:include schemaLocation="http://docs.oasis-open.org/security/saml/v2.0/saml-schema-authn-
context-types-2.0.xsd"/>
  <!-- new type definitions for different GBA procedures -->
  <xs:complexType name="GBAMechanismTypeType">
    <xs:annotation>
      <xs:documentation>
        GBA mechanism used in the bootstrapping procedure.
      </xs:documentation>
    </xs:annotation>
  </xs:complexType>
  <xs:element name="LegacyGBA" type="GBAMechanismTypeType">
    <xs:annotation>
      <xs:documentation>
        Legacy GBA where existing old authentication
        frameworks are used for bootstrapping such
        2G GBA in 3GPP, and CDMA 1x and CDMA 1xEvDo
        in 3GPP2.
      </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="GBAMobileMobile" type="GBAMechanismTypeType">
    <xs:annotation>
      <xs:documentation>
        GBA using AKA as specified in 3GPP and 3GPP2.
        The shared secret is derived in the mobile and
        used in the mobile.
      </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="GBAUICCMobile" type="GBAMechanismTypeType">
    <xs:annotation>
      <xs:documentation>
        GBA_U using AKA as specified in 3GPP and 3GPP2.
        The shared secret is derived in the UICC and used
        in the mobile.
      </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="GBAUICCUICC" type="GBAMechanismTypeType">
    <xs:annotation>
      <xs:documentation>
        GBA_U using AKA as specified in 3GPP and 3GPP2.
        The shared secret is derived in the UICC and used
        in the UICC.
      </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:complexType name="GBAMechanismType">
    <xs:choice>
      <xs:element ref="LegacyGBA"/>
      <xs:element ref="GBAMobileMobile"/>
      <xs:element ref="GBAUICCMobile"/>
      <xs:element ref="GBAUICCUICC"/>
    </xs:choice>
  </xs:complexType>
  <!-- new element for GBA authentication context classes -->
  <xs:element name="GBAMechanism" type="GBAMechanismType"/>
</xs:schema>
```

F.4 GBA authentication context declaration classes

F.4.1 GBAOneFactorUnregistered

This class reflects that there were no mobile customer registration procedures and an authentication of the UE is done without requiring explict end-user interaction.

F.4.1.1 Associated 3GPP URI

urn:3gpp:gba:ac:saml:2007-08:classes:GBAOneFactorUnregistered

F.4.1.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns="urn:3gpp:gba:ac:saml:2007-08:classes:GBAOneFactorUnregistered"
  targetNamespace="urn:3gpp:gba:ac:saml:2007-08:classes:GBAOneFactorUnregistered"
  finalDefault="extension"
  version="1.0">
  <xs:redefine schemaLocation="3gpp-gba-saml-authn-context-types-1.0.xsd">
    <xs:annotation>
      <xs:documentation>
        The GBA authentication context class comparable to
        the SAML MobileOneFactorUnregistered class.
        Id: 3gpp-gba-saml-authn-context-gbaonefactor-unreg-1.0.xsd
      </xs:documentation>
    </xs:annotation>
    <xs:complexType name="AuthnContextDeclarationBaseType">
      <xs:complexContent>
        <xs:restriction base="AuthnContextDeclarationBaseType">
          <xs:sequence>
            <xs:element ref="Identification" minOccurs="0"/>
            <xs:element ref="TechnicalProtection" minOccurs="0"/>
            <xs:element ref="OperationalProtection" minOccurs="0"/>
            <xs:element ref="AuthnMethod"/>
            <xs:element ref="GoverningAgreements" minOccurs="0"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="ID" type="xs:ID" use="optional"/>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="AuthnMethodBaseType">
      <xs:complexContent>
        <xs:extension base="AuthnMethodBaseType">
          <xs:sequence>
            <xs:element ref="GBAMechanism" minOccurs="0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
      <!-- difference compared to SAML: GBAMechanism -->
    </xs:complexType>
    <xs:complexType name="AuthenticatorBaseType">
      <xs:complexContent>
        <xs:restriction base="AuthenticatorBaseType">
          <xs:sequence>
            <xs:choice>
              <xs:element ref="SharedSecretChallengeResponse"/>
            </xs:choice>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only shared secret -->
    </xs:complexType>
    <xs:complexType name="AuthenticatorTransportProtocolType">
      <xs:complexContent>
        <xs:restriction base="AuthenticatorTransportProtocolType">
          <xs:sequence>
```

```
          <xs:choice>
            <xs:element ref="HTTP"/>
            <xs:element ref="SSL"/>
          </xs:choice>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only HTTP (Digest) or SSL (PSK TLS) -->
  </xs:complexType>
  <xs:complexType name="OperationalProtectionType">
    <xs:complexContent>
      <xs:restriction base="OperationalProtectionType">
        <xs:sequence>
          <xs:element ref="SecurityAudit"/>
          <xs:element ref="DeactivationCallCenter"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="TechnicalProtectionBaseType">
    <xs:complexContent>
      <xs:restriction base="TechnicalProtectionBaseType">
        <xs:sequence>
          <xs:choice>
            <xs:element ref="SecretKeyProtection"/>
          </xs:choice>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only secret key (no private key) -->
  </xs:complexType>
  <!-- no PrivateKeyProtectionType -->
  <xs:complexType name="SecretKeyProtectionType">
    <xs:complexContent>
      <xs:restriction base="SecretKeyProtectionType">
        <xs:sequence>
          <xs:element ref="KeyStorage"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="KeyStorageType">
    <xs:complexContent>
      <xs:restriction base="KeyStorageType">
        <xs:attribute name="medium" use="required">
          <xs:simpleType>
            <xs:restriction base="mediumType">
              <xs:enumeration value="MobileDevice"/>
              <xs:enumeration value="smartcard"/>
              <!-- MobileDevice: GBA_ME Ks_NAF, GBA_U Ks_ext_NAF -->
              <!-- smartcard: GBA_U Ks_int_NAF -->
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only mobile dev, smartcard -->
  </xs:complexType>
  <xs:complexType name="SecurityAuditType">
    <xs:complexContent>
      <xs:restriction base="SecurityAuditType">
        <xs:sequence>
          <xs:element ref="SwitchAudit"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="IdentificationType">
    <xs:complexContent>
      <xs:restriction base="IdentificationType">
```

```
          <xs:sequence>
            <xs:element ref="GoverningAgreements"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="nym">
            <xs:simpleType>
              <xs:restriction base="nymType">
                <xs:enumeration value="anonymity"/>
                <xs:enumeration value="pseudonymity"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
  </xs:redefine>
</xs:schema>
```

F.4.2 GBATwoFactorUnregistered

This class reflects that there were no mobile customer registration procedures and a two-factor based authentication of the UE is done requiring an explicit end-user interaction during authentication procedure (e.g., a PIN needs to be typed).

F.4.2.1 Associated 3GPP URI

urn:3gpp:gba:ac:saml:2007-08:classes:GBATwoFactorUnregistered

F.4.2.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns="urn:3gpp:gba:ac:saml:2007-08:classes:GBATwoFactorUnregistered"
  targetNamespace="urn:3gpp:gba:ac:saml:2007-08:classes:GBATwoFactorUnregistered"
  finalDefault="extension"
  version="1.0">
  <xs:redefine schemaLocation="3gpp-gba-saml-authn-context-types-1.0.xsd">
    <xs:annotation>
      <xs:documentation>
        The GBA authentication context class comparable to
        the SAML MobileTwoFactorUnregistered class.
        Id: 3gpp-gba-saml-authn-context-gbatwofactor-unreg-1.0.xsd
      </xs:documentation>
    </xs:annotation>
    <xs:complexType name="AuthnContextDeclarationBaseType">
      <xs:complexContent>
        <xs:restriction base="AuthnContextDeclarationBaseType">
          <xs:sequence>
            <xs:element ref="Identification" minOccurs="0"/>
            <xs:element ref="TechnicalProtection" minOccurs="0"/>
            <xs:element ref="OperationalProtection" minOccurs="0"/>
            <xs:element ref="AuthnMethod"/>
            <xs:element ref="GoverningAgreements" minOccurs="0"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="ID" type="xs:ID" use="optional"/>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="AuthnMethodBaseType">
      <xs:complexContent>
        <xs:extension base="AuthnMethodBaseType">
          <xs:sequence>
            <xs:element ref="GBAMechanism" minOccurs="0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
      <!-- difference compared to SAML: GBAMechanism -->
    </xs:complexType>
```

```
    <xs:complexType name="AuthenticatorBaseType">
      <xs:complexContent>
        <xs:restriction base="AuthenticatorBaseType">
          <xs:sequence>
            <xs:choice>
              <xs:element ref="SharedSecretChallengeResponse"/>
            </xs:choice>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only shared secret -->
    </xs:complexType>
    <xs:complexType name="AuthenticatorTransportProtocolType">
      <xs:complexContent>
        <xs:restriction base="AuthenticatorTransportProtocolType">
          <xs:sequence>
            <xs:choice>
              <xs:element ref="HTTP"/>
              <xs:element ref="SSL"/>
            </xs:choice>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only HTTP (Digest) or SSL (PSK TLS) -->
    </xs:complexType>
    <xs:complexType name="OperationalProtectionType">
      <xs:complexContent>
        <xs:restriction base="OperationalProtectionType">
          <xs:sequence>
            <xs:element ref="SecurityAudit"/>
            <xs:element ref="DeactivationCallCenter"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="TechnicalProtectionBaseType">
      <xs:complexContent>
        <xs:restriction base="TechnicalProtectionBaseType">
          <xs:sequence>
            <xs:choice>
              <xs:element ref="SecretKeyProtection"/>
            </xs:choice>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only secret key (no private key) -->
    </xs:complexType>
    <!-- no PrivateKeyProtectionType -->
    <xs:complexType name="SecretKeyProtectionType">
      <xs:complexContent>
        <xs:restriction base="SecretKeyProtectionType">
          <xs:sequence>
            <xs:element ref="KeyActivation"/>
            <xs:element ref="KeyStorage"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: storage, activation mandatory -->
    </xs:complexType>
    <xs:complexType name="KeyActivationType">
      <xs:complexContent>
        <xs:restriction base="KeyActivationType">
          <xs:sequence>
            <xs:element ref="ActivationPin"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- not explicitely redefined by corresponding class in SAML -->
    </xs:complexType>
    <xs:complexType name="KeyStorageType">
      <xs:complexContent>
```

```
        <xs:restriction base="KeyStorageType">
          <xs:attribute name="medium" use="required">
            <xs:simpleType>
              <xs:restriction base="mediumType">
                <xs:enumeration value="MobileDevice"/>
                <xs:enumeration value="smartcard"/>
                <!-- MobileDevice: GBA_ME Ks_NAF, GBA_U Ks_ext_NAF -->
                <!-- smartcard: GBA_U Ks_int_NAF -->
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only mobile dev, smartcard -->
    </xs:complexType>
    <xs:complexType name="SecurityAuditType">
      <xs:complexContent>
        <xs:restriction base="SecurityAuditType">
          <xs:sequence>
            <xs:element ref="SwitchAudit"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="IdentificationType">
      <xs:complexContent>
        <xs:restriction base="IdentificationType">
          <xs:sequence>
            <xs:element ref="GoverningAgreements"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="nym">
            <xs:simpleType>
              <xs:restriction base="nymType">
                <xs:enumeration value="anonymity"/>
                <xs:enumeration value="pseudonymity"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
  </xs:redefine>
</xs:schema>
```

F.4.3 GBAOneFactorContract

This class reflects that mobile customer registration procedures have taken place and an authentication of the UE is done without requiring explicit end-user interaction.

F.4.3.1 Associated 3GPP URI

urn:3gpp:gba:ac:saml:2007-08:classes:GBAOneFactorContract

F.4.3.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns="urn:3gpp:gba:ac:saml:2007-08:classes:GBAOneFactorContract"
  targetNamespace="urn:3gpp:gba:ac:saml:2007-08:classes:GBAOneFactorContract"
  finalDefault="extension"
  version="1.0">
  <xs:redefine schemaLocation="3gpp-gba-saml-authn-context-types-1.0.xsd">
    <xs:annotation>
      <xs:documentation>
        The GBA authentication context class comparable to
        the SAML MobileOneFactorContract class.
```

```
      Id: 3gpp-gba-saml-authn-context-gbaonefactor-reg-1.0.xsd
    </xs:documentation>
  </xs:annotation>
  <xs:complexType name="AuthnContextDeclarationBaseType">
    <xs:complexContent>
      <xs:restriction base="AuthnContextDeclarationBaseType">
        <xs:sequence>
          <xs:element ref="Identification" minOccurs="0"/>
          <xs:element ref="TechnicalProtection" minOccurs="0"/>
          <xs:element ref="OperationalProtection" minOccurs="0"/>
          <xs:element ref="AuthnMethod"/>
          <xs:element ref="GoverningAgreements" minOccurs="0"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:ID" use="optional"/>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="AuthnMethodBaseType">
    <xs:complexContent>
      <xs:extension base="AuthnMethodBaseType">
        <xs:sequence>
          <xs:element ref="GBAMechanism" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
    <!-- difference compared to SAML: GBAMechanism -->
  </xs:complexType>
  <xs:complexType name="AuthenticatorBaseType">
    <xs:complexContent>
      <xs:restriction base="AuthenticatorBaseType">
        <xs:sequence>
          <xs:choice>
            <xs:element ref="SharedSecretChallengeResponse"/>
          </xs:choice>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only shared secret -->
  </xs:complexType>
  <xs:complexType name="AuthenticatorTransportProtocolType">
    <xs:complexContent>
      <xs:restriction base="AuthenticatorTransportProtocolType">
        <xs:sequence>
          <xs:choice>
            <xs:element ref="HTTP"/>
            <xs:element ref="SSL"/>
          </xs:choice>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only HTTP (Digest) or SSL (PSK TLS) -->
  </xs:complexType>
  <xs:complexType name="OperationalProtectionType">
    <xs:complexContent>
      <xs:restriction base="OperationalProtectionType">
        <xs:sequence>
          <xs:element ref="SecurityAudit"/>
          <xs:element ref="DeactivationCallCenter"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="TechnicalProtectionBaseType">
    <xs:complexContent>
      <xs:restriction base="TechnicalProtectionBaseType">
        <xs:sequence>
          <xs:choice>
            <xs:element ref="SecretKeyProtection"/>
          </xs:choice>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
```

```
      </xs:complexContent>
      <!-- difference compared to SAML: only secret key (no private key) -->
    </xs:complexType>
    <!-- no PrivateKeyProtectionType -->
    <xs:complexType name="SecretKeyProtectionType">
      <xs:complexContent>
        <xs:restriction base="SecretKeyProtectionType">
          <xs:sequence>
            <xs:element ref="KeyStorage"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="KeyStorageType">
      <xs:complexContent>
        <xs:restriction base="KeyStorageType">
          <xs:attribute name="medium" use="required">
            <xs:simpleType>
              <xs:restriction base="mediumType">
                <xs:enumeration value="MobileDevice"/>
                <xs:enumeration value="smartcard"/>
                <!-- MobileDevice: GBA_ME Ks_NAF, GBA_U Ks_ext_NAF -->
                <!-- smartcard: GBA_U Ks_int_NAF -->
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only mobile dev, smartcard -->
    </xs:complexType>
    <xs:complexType name="SecurityAuditType">
      <xs:complexContent>
        <xs:restriction base="SecurityAuditType">
          <xs:sequence>
            <xs:element ref="SwitchAudit"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="IdentificationType">
      <xs:complexContent>
        <xs:restriction base="IdentificationType">
          <xs:sequence>
            <xs:element ref="GoverningAgreements"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="nym">
            <xs:simpleType>
              <xs:restriction base="nymType">
                <xs:enumeration value="anonymity"/>
                <xs:enumeration value="verinymity"/>
                <xs:enumeration value="pseudonymity"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
  </xs:redefine>
</xs:schema>
```

F.4.4 GBATwoFactorContract

This class reflects that mobile customer registration procedures have taken place and a two-factor based authentication of the UE is done requiring an explicit end-user interaction during authentication procedure (e.g., a PIN needs to be typed).

F.4.4.1 Associated 3GPP URI

urn:3gpp:gba:ac:saml:2007-08:classes:GBATwoFactorContract

F.4.4.2 Class schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns="urn:3gpp:gba:ac:saml:2007-08:classes:GBATwoFactorContract"
  targetNamespace="urn:3gpp:gba:ac:saml:2007-08:classes:GBATwoFactorContract"
  finalDefault="extension"
  version="1.0">
  <xs:redefine schemaLocation="3gpp-gba-saml-authn-context-types-1.0.xsd">
    <xs:annotation>
      <xs:documentation>
        The GBA authentication context class comparable to
        the SAML MobileTwoFactorContract class.
        Id: 3gpp-gba-saml-authn-context-mobiletwofactor-reg-1.0.xsd
      </xs:documentation>
    </xs:annotation>
    <xs:complexType name="AuthnContextDeclarationBaseType">
      <xs:complexContent>
        <xs:restriction base="AuthnContextDeclarationBaseType">
          <xs:sequence>
            <xs:element ref="Identification" minOccurs="0"/>
            <xs:element ref="TechnicalProtection" minOccurs="0"/>
            <xs:element ref="OperationalProtection" minOccurs="0"/>
            <xs:element ref="AuthnMethod"/>
            <xs:element ref="GoverningAgreements" minOccurs="0"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="ID" type="xs:ID" use="optional"/>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
    <xs:complexType name="AuthnMethodBaseType">
      <xs:complexContent>
        <xs:extension base="AuthnMethodBaseType">
          <xs:sequence>
            <xs:element ref="GBAMechanism" minOccurs="0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
      <!-- difference compared to SAML: GBAMechanism -->
    </xs:complexType>
    <xs:complexType name="AuthenticatorBaseType">
      <xs:complexContent>
        <xs:restriction base="AuthenticatorBaseType">
          <xs:sequence>
            <xs:choice>
              <xs:element ref="SharedSecretChallengeResponse"/>
            </xs:choice>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only shared secret -->
    </xs:complexType>
    <xs:complexType name="AuthenticatorTransportProtocolType">
      <xs:complexContent>
        <xs:restriction base="AuthenticatorTransportProtocolType">
          <xs:sequence>
            <xs:choice>
              <xs:element ref="HTTP"/>
              <xs:element ref="SSL"/>
            </xs:choice>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
      <!-- difference compared to SAML: only HTTP (Digest) or SSL (PSK TLS) -->
    </xs:complexType>
    <xs:complexType name="OperationalProtectionType">
      <xs:complexContent>
```

```
      <xs:restriction base="OperationalProtectionType">
        <xs:sequence>
          <xs:element ref="SecurityAudit"/>
          <xs:element ref="DeactivationCallCenter"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="TechnicalProtectionBaseType">
    <xs:complexContent>
      <xs:restriction base="TechnicalProtectionBaseType">
        <xs:sequence>
          <xs:choice>
            <xs:element ref="SecretKeyProtection"/>
          </xs:choice>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only secret key (no private key) -->
  </xs:complexType>
  <!-- no PrivateKeyProtectionType -->
  <xs:complexType name="SecretKeyProtectionType">
    <xs:complexContent>
      <xs:restriction base="SecretKeyProtectionType">
        <xs:sequence>
          <xs:element ref="KeyActivation"/>
          <xs:element ref="KeyStorage"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: storage, activation mandatory -->
  </xs:complexType>
  <xs:complexType name="KeyActivationType">
    <xs:complexContent>
      <xs:restriction base="KeyActivationType">
        <xs:sequence>
          <xs:element ref="ActivationPin"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- not explicitely redefined by corresponding class in SAML -->
  </xs:complexType>
  <xs:complexType name="KeyStorageType">
    <xs:complexContent>
      <xs:restriction base="KeyStorageType">
        <xs:attribute name="medium" use="required">
          <xs:simpleType>
            <xs:restriction base="mediumType">
              <xs:enumeration value="MobileDevice"/>
              <xs:enumeration value="smartcard"/>
              <!-- MobileDevice: GBA_ME Ks_NAF, GBA_U Ks_ext_NAF -->
              <!-- smartcard: GBA_U Ks_int_NAF -->
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:restriction>
    </xs:complexContent>
    <!-- difference compared to SAML: only mobile dev, smartcard -->
  </xs:complexType>
  <xs:complexType name="SecurityAuditType">
    <xs:complexContent>
      <xs:restriction base="SecurityAuditType">
        <xs:sequence>
          <xs:element ref="SwitchAudit"/>
          <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
    <!-- no difference compared to SAML -->
  </xs:complexType>
  <xs:complexType name="IdentificationType">
    <xs:complexContent>
      <xs:restriction base="IdentificationType">
```

```
          <xs:sequence>
            <xs:element ref="GoverningAgreements"/>
            <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="nym">
            <xs:simpleType>
              <xs:restriction base="nymType">
                <xs:enumeration value="anonymity"/>
                <xs:enumeration value="verinymity"/>
                <xs:enumeration value="pseudonymity"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:restriction>
      </xs:complexContent>
      <!-- no difference compared to SAML -->
    </xs:complexType>
  </xs:redefine>
</xs:schema>
```

Annex F (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2004-09	CN#25	NP-040410			Version 2.0.0 approved in CN#25	2.0.0	6.0.0
2004-12	CN#26	NP-040580	001		Authorization Flag Code Annex	6.0.0	6.1.0
2004-12			002		Finalization of GAA Service Identifier		
2004-12			003	1	BSF control information (bsfInfo) tag to GUSS		
2004-12			005		Structure to GAA Service Indentifier		
2004-12			006	1	Finalisation of terminology		
2004-12			008	1	Command code 310 Zn messages		
2004-12			009		Introduction of NAF groups		
2005-01					Fix Word problem	6.1.0	6.1.1
2005-03	CN#27	NP-050041	010		GAA Error Codes	6.1.1	6.2.0
			011		Only one AV from HSS to BSF		
			012		Clarification of LifeTime/ExpiryTime terminology		
			013	1	Application identifiers to Z-interfaces		
			014	1	Modification of key lifetime material		
2005-06	CT#28	CP-050090	0015		XML extensibility	6.2.0	6.3.0
			0016	1	Remove BSF from visited network		
2005-09	CT#29	CP-050300	0017	1	Correction for GBA with multiple HSS"s	6.3.0	6.4.0
			0021		Key naming alignment with TS 33.220		
2005-09	CT#29	CP-050317	0019	1	Key indication in USS	6.4.0	7.0.0
			0020	2	Addition of GUSS timestamp to Zh reference point		
2005-12	CT#30	CP-050613	0019	1	XML syntax correction	7.0.0	7.1.0
		CP-050624	0022	1	2G GBA implementation to Zh and Zn		
2006-03	CT#31	CP-060071	0025	1	Correction of NAF_ID	7.1.0	7.2.0
			0026	1	HTTP based Zn interface		
			0027	1	Liberty authentication context for GAA		
2006-06	CT#32	CP-060318	0028	1	Optionality of the HTTP based Zn interface for NAF	7.2.0	7.3.0
			0029	1	Replacement of "GAA-Application-Type-Code" with "GAA Service Type Code"		
2006-09	CT#33	CP-060415	0030	1	Inconsistent description about the proxy between BSF and visted NAF	7.3.0	7.4.0
			0031		Only one instance of SIP-Auth-Data-Item sent in MAA		
2006-12	CT#34	CP-060567	0032	1	Zh MAR/MAA Diameter Application identifier	7.4.0	7.5.0
			0033		Definition of GAA Service Type Code for Liberty Alliance Interworking		
			0034		Corrections to GAA authentication context schema		
			0035		WSDL definition enhancements for Zn interface		
2007-06	CT#36	CP-070320	0037	1	Addition of Service Type For Rel-7	7.5.0	7.6.0
		CP-070320	0038	1	Bug fixes on Zn WSDL definition		
		CP-070317	0040		GUSS schema fix		
2007-09	CT#37	CP-070528	0036	4	HLR support for BSF	7.6.0	7.7.0
			0041	1	Addition of SAML authentication context declaration classes		
2007-12	CT#38	CP-070741	0042	1	GUSS and USS handling corrections	7.7.0	7.8.0
2008-03	CT#39	CP-080013	0045		GBA-UserSecSettings XML fix	7.8.0	7.9.0
2008-06	CT#40	CP-080275	0047	2	Key choice correction	7.9.0	7.10.0
2008-12	CT#42				Upgraded unchanged from Rel-7	7.10.0	8.0.0
2008-03	CT#43	CP-090033	0048		Introduction of GBA Push within TS 29.109 – Scope, References and Definitions	8.0.0	8.1.0
			0049	2	Introduction of GBA Push within TS 29.109 – Zpn Interface		
			0050	1	Introduction of GBA Push within TS 29.109 – Impacts on Zh Interface		
			0051	3	User identity to HSS Resolution mechanism using a Diameter Proxy		
			0055	1	Renaming of the reference point between BSF and HLR		

History

Document history		
V8.0.0	January 2009	Publication
V8.1.0	April 2009	Publication